



Intel® Ethernet Controller X710- TM4/AT2 Datasheet

NEX Cloud Networking Group (NCNG)

Doc. No.: 596333-005
Revision: 2.4
November 2022



No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

This document (and any related software) is Intel copyrighted material, and your use is governed by the express license under which it is provided to you. Unless the license provides otherwise, you may not use, modify, copy, publish, distribute, disclose or transmit this document (and related materials) without Intel's prior written permission. This document (and related materials) is provided as is, with no express or implied warranties, other than those that are expressly stated in the license.

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.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

Other names and brands may be claimed as the property of others.

© 2019-2022 Intel Corporation.



Revision History

Revision	Date	Notes
2.4	November 2022	Updated corrupted graphic in Figure 13-15 , "Mechanical Package Diagram (Top/Side View)".
2.3	April 2022	<p>Sections revised:</p> <ul style="list-style-type: none"> 2.2.8 (Testability and Debug Pins) 3.2.1.1 (MAC Speed Configuration) 3.2.2 (Internal PHY for 10GBASE-T Ports) <p>Tables revised:</p> <ul style="list-style-type: none"> 2.3 (Ethernet Interface Pins) 2.11 (Miscellaneous Pins) 2.12 (Testability and Debug Pins) 2.13 (Reserved and No-connect Pins) 2.14 (No-connect Pins X710-AT2/V710-AT2) 2.16 (Power Supply Pins) 2.17 (Pull-up and Pull-down Resistors) 9-74 (Controller information items)
2.2	February 2022	<p>Global change:</p> <ul style="list-style-type: none"> Changed "IEEE802.3" and "IEEE 802.3" to "IEEE Std. 802.3-2018". Changed "IEEE802.3xx" and IEEE 802.3xx" to "IEEE Std. 802.3xx".
2.1	January 2021	Added Section 1.1.8.1 (Protect, Detect and Recover).
2.0	August 2019	Initial Release (Intel Public).



NOTE: *This page intentionally left blank.*



1.0 Introduction

This document describes the external architecture (including device operation, pin descriptions, register definitions, etc.) for the X710-TM4, a quad-port integrated 10GBASE-T and 10 Gigabit Ethernet (GbE) serial single-package controller. The X710-TM4 can connect up to four Ethernet ports: two 100BASE-TX/1000BASE-T/2.5GBASE-T/5GBASE-T/10GBASE-T and two 10GBASE-KR/SFI ports. It reflects the silicon device capability while the *Intel® Ethernet Controller X710-TM4 Feature Support Matrix* lists the features and interfaces actually supported in the NVM and software (<http://www.intel.com/content/www/us/en/embedded/products/networking/xl710-ethernetcontroller-feature-matrix.html>).

This document is intended as a reference for architects, logic designers, firmware and software device driver developers, board designers, test engineers, or anyone else who might need specific technical or programming information about the X710-TM4.

The X710-TM4 combines Network Function Virtualization (NFV) and network virtualization overlays, server virtualization and unified networking into a single silicon device. It is designed to address the target markets listed in [Table 1-1](#).

Note: The X710-TM4 does NOT support Fibre Channel over Ethernet (FCoE).

Table 1-1. X710-TM4 Target Markets

Description
<p>Enterprise Networking — The X710-TM4 strengths in this market are networking performance, energy efficiency, support for OEM-specific requirements, broad Operating System (OS) and Virtual Machine Monitor (VMM) support, automation (including resource provisioning and monitoring, and workload balancing), converged networking, and emerging standards. For example: Data Center Bridging (DCB) and Virtual Bridging (VEB).</p>
<p>Cloud Networking — In the emerging cloud networking market, where computing infrastructure and software are sold as services, and where the large data centers of Internet portal companies such as Google*, Microsoft* and Amazon* drive unique requirements, the X710-TM4 has these strengths: networking performance, energy efficiency, automation (including resource provisioning and monitoring, and workload balancing), sophisticated packet header parsing, and quality open source drivers. In cases where computing infrastructure is sold as a service, the X710-TM4 features importance in Enterprise networking as well as High Performance Computing (HPC) importance to the cloud.</p>
<p>Embedded — The embedded market includes security appliances, networking and telecom products, storage targets, etc. The X710-TM4 strengths in this market are performance, broad operating system support, quality open source drivers and Intel product support.</p>

1.1 Block Diagram

Figure 1-1 and Figure 1-2 show the X710-TM4’s block architecture. Figure 1-3 shows the PHY/module connectivity. This section also provides an overview of the X710-TM4’s external interfaces and top-level internal blocks.

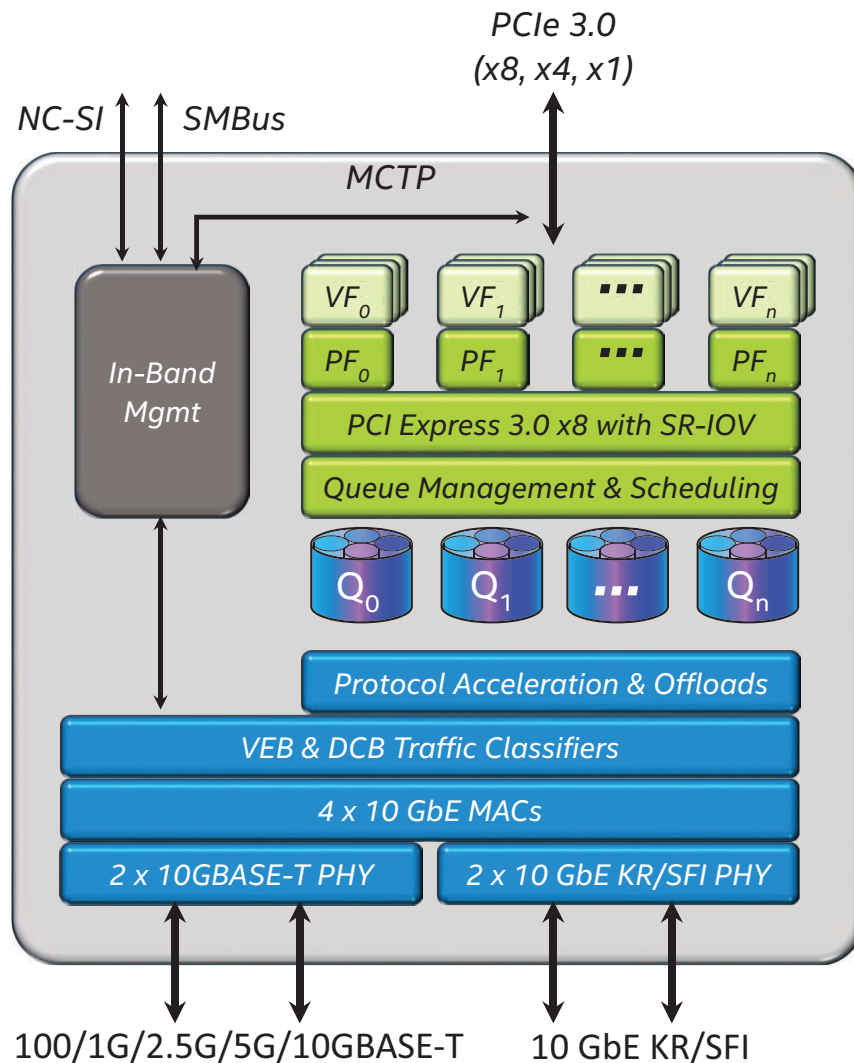


Figure 1-1. X710-TM4 Block Quad-port Diagram

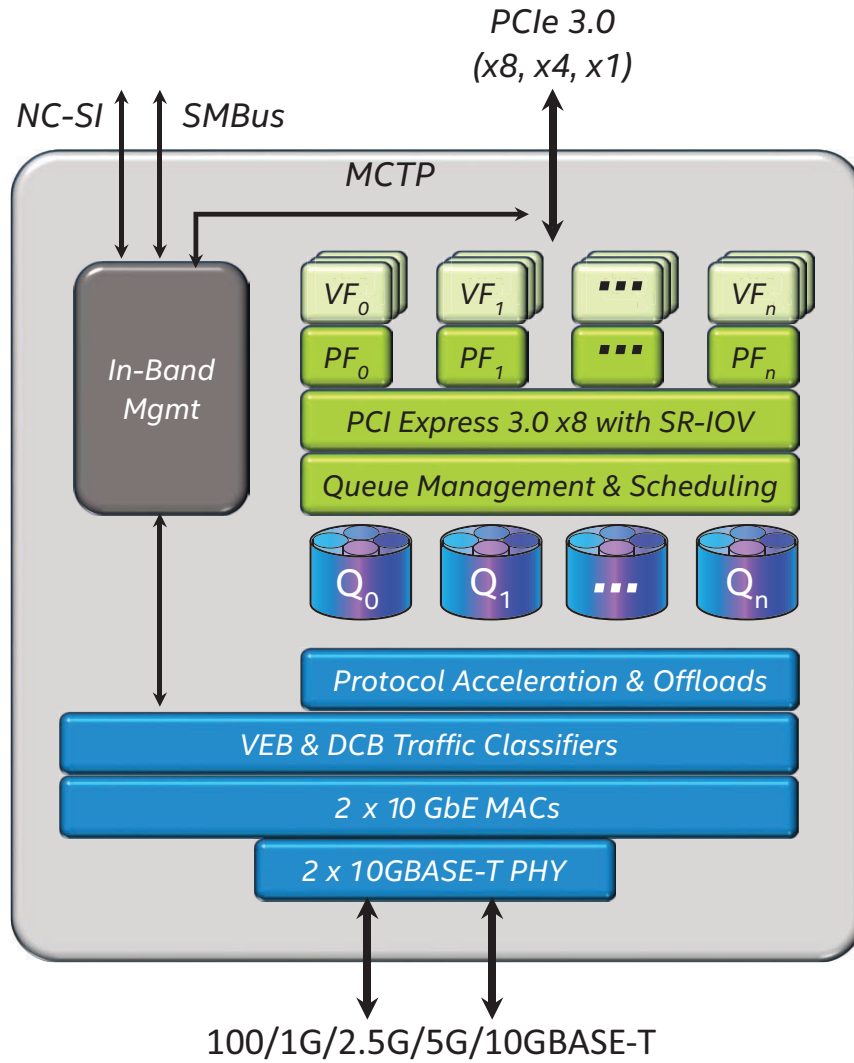


Figure 1-2. X710-AT2 Block Dual-port Diagram

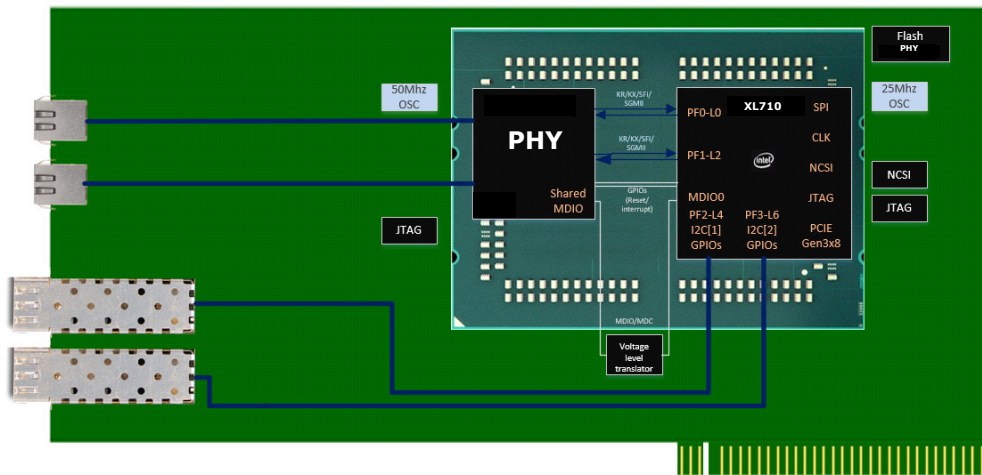


Figure 1-3. PHY/Module Connectivity (Internal PHY Connection Shown)

1.1.1 PCI Express* (PCIe*) with Single Root I/O Virtualization (SR-IOV)

The X710-TM4 implements a PCIe v3.0 x8 host interface, which operates at up to 8 GT/s or 64 Gb/s. See [Section 2.2.1](#) for a full pin description and [Section 13.6.3](#) for interface timing characteristics.

The X710-TM4’s PCIe host interface implements up to 16 Physical Functions (PFs), and up to 128 Virtual Functions (VFs). More details on the X710-TM4’s PCIe features are provided in [Section 1.2](#). [Section 11.0](#) describes the PCIe programming interface.

1.1.2 Host Memory Objects

The X710-TM4 operating system drivers set up a wide variety of host memory objects that are comprehended and manipulated by the X710-TM4. All objects are set up in the context of a PCI function. This is important for at least two reasons:

- Platform security. For example, many types of the X710-TM4 host memory objects are privileged, and are only allowed to be set up in the context of a PF. For example, they are dis-allowed in the context of a Virtual Function (VF), which operates at a lower privilege level than a PF.
- Reliability. If the operating system resets a PCI function, that function’s host memory objects are lost, but the host memory objects of other PCI functions survive.



Following are three types of memory objects:

- LAN Queue Pairs (LQPs). These are ring buffers (one transmit, one receive) for submitting commands to the Local Area Network (LAN) engine. Commands take the form of packets/data to be transmitted, descriptors for empty host memory buffers to be filled with received packets/data, etc. LQPs are typically mapped into the operating system kernel space. In a virtualized server, they can be assigned either to the VMM, or to VMs using SR-IOV. The X710-TM4 supports up to 1536 LQPs that can be assigned to PFs or VFs as needed. The LQPs assigned to a particular PCI function can be used in these important ways:
 - For distributing packet processing work to the different processors in a multi-processor system. On the transmit side, this is done by simply dedicating an independent transmit queue for each CPU to use. On the receive side, packets are classified by the X710-TM4 under operating system control into groups of conversations. Each group of conversations is assigned its own receive queue and receiving processor. Microsoft* Receive Side Scaling (RSS) is one popular example of this method.
 - For assigning Traffic Class (TC). Transmit queues assigned to different TCs are serviced at different rates by the X710-TM4 transmit scheduler. Receive queues assigned to different TCs can be serviced at different rates by a Quality of Service (QoS) operating system and its software device drivers.
- Admin Queue Pairs. Each PCI function maps an Admin Queue Pair (AQP): one Admin Send Queue (ASQ) and one Admin Receive Queue (ARQ). The ASQ is a ring buffer used by the host driver for submitting commands to configure the X710-TM4. Commands submitted on the ASQ are serviced by X710-TM4's Embedded Management Processor (EMP). Some examples are commands to reconfigure: the Tx scheduler, an Ethernet link, various internal switch and DCB settings, etc. The ARQ conveys events from the EMP to the host driver that are not an immediate result of an ASQ command. The host driver posts empty buffers to the ARQ and the EMP fills them with events. Note that commands submitted on a VF ASQ are typically not directly serviced by the EMP, but rather are redirected to the associated PF ARQ. This enables the PF driver to inspect and authorize all VF ASQ commands that are often of a privileged nature. For more details on AQPs, see [Section 7.10](#).

1.1.3 Ethernet Media Access Controller (MAC) and PHY

The X710-TM4 integrates four IEEE Std 802.3 compliant Ethernet MACs. All MACs support transmitting and receiving jumbo frames up to 9728 bytes and 802.3x flow control frames or 802.3bd priority-based flow control frames. See [Section 3.2.1](#) for details.

- MAC 0 and MAC 1 connect to an integrated BASE-T PHY that supports 10GBASE-TX and 1G/2.5G/5G/10GBASE-T interfaces
- MAC 2 and MAC 3 connect to an integrated serial PHY that supports:
 - 1000BASE-X and 10GBASE-KR backplane Ethernet
 - SFP+ pluggable module applications supporting twinax copper and optical modules

The X710-TM4 supports up to four active Ethernet ports and can be configured to support different levels of Ethernet PHY integration using various IEEE standard interfaces.

Note: The two serial PHY ports can also be connected to an external BASE-T PHY to implement a quad-port BASE-T solution.



The X710-TM4 implements either two independent Management Data Input/Output (MDIO) interfaces or two independent Inter-integrated Circuit (I²C) interfaces for connection to external PHYs. These enable host software or the X710-TM4 firmware to control connected external PHYs, including the ability to read and write PHY registers. Details on how they are typically connected and used are described in [Section 3.2.10.1](#) through [Section 3.2.10.5](#).

1.1.4 Transmit Scheduler

The X710-TM4 provides management interfaces that enable each LAN transmit queue to be placed into a queue set. A queue set is a list of transmit queues that belong to the same Traffic Class (TC) and are treated equally by the X710-TM4 transmit scheduler. The X710-TM4 supports a maximum of 384 Virtual Station Interfaces (VSIs), and an average of two queue sets per VSI. There is no limit to the number of transmit queues that can belong to a queue set.

Typically, one or more queue sets are assigned to a PCI function/VSI, according to the number of TCs it uses. For LAN, the queue sets are often only one or a small handful of transmit queues, corresponding to the number of host CPUs assigned to generate traffic on that TC.

The X710-TM4 transmit scheduler supports independent programming of a wide variety of controls that affect queue set behavior. Each queue set can be programmed with an independent static rate limit. Each group of queue sets assigned to a PCI function/VSI can be programmed with DCB Enhanced Transmission Selection (ETS) settings, group static rate limit, and uplink bandwidth share.

The X710-TM4 transmit scheduler also implements controls similar to those previously described for the various internal resources that comprise the X710-TM4's hierarchy of internal switching components. These internal resources include Virtual Ethernet Bridge (VEB), Virtual Ethernet Port Aggregator (VEPA) and SComp v-ports, as well as the physical Ethernet ports of the device.

For more detail on the X710-TM4 transmit scheduler features and operation, see [Section 7.8](#).

1.1.5 Host Memory Cache (HMC)

The X710-TM4 LAN engine uses host memory as a backing store for a variety of context objects. The HMC is responsible for caching and managing these context objects as follows examples:

- The LAN engine uses two HMC context objects per Queue Pair (QP), one for the Transmit Queue (TQ) and one for the Receive Queue (RQ). Parameters in the TQ context include a Transmit Segmentation Offload (TSO) state that is detailed in [Section 8.4.3.4](#). Parameters in the RQ context include a queue base address and associated pointers that are detailed in [Section 8.3.3.2](#).

General information on HMC operation and configuration is provided in [Section 7.9](#).

1.1.6 LAN Engine

The LAN engine implements the host programming interface for traditional LAN traffic in both virtualized and non-virtualized scenarios. The X710-TM4 implements 384 VSIs (virtual-NICs) used to distribute traffic to PCI physical functions and virtual functions. These VSIs can connect to the host via 1536 LAN QPs.



The LAN engine also implements all of the X710-TM4's performance optimizations for traditional LAN traffic. This includes packet checksum offloads, Transmission Control Protocol/User Datagram Protocol (TCP/UDP Segmentation Offload), RSS and others.

1.1.7 System Management

The X710-TM4 participates in system management by providing networking services to platform Management Controllers (MCs); also called Baseboard Management Controllers (BMCs). The X710-TM4 is also accessible to these devices to be managed as any platform resource that is managed by the system.

Networking services are provided through the Pass-through (PT) functionality described in [Section 9.1](#). Several sideband channels are provided to connect to an MC:

- System Management Bus (SMBus)
- Network Controller Sideband Interface (NC-SI)
- PCIe; together with Management Component Transport Protocol (MCTP)

The X710-TM4 also supports communication between local software agents and the local MC through an operating system-to-BMC capability described in [Section 9.4](#).

1.1.8 Embedded Management Processor (EMP)

The EMP handles all management duties that cannot be performed by the X710-TM4's device drivers and must be carried out on-chip. This includes performing parts of the X710-TM4 power-on sequence, handling AQ commands, initializing the X710-TM4 Ethernet ports, participating in various fabric configuration protocols such as DCBX and other Link Layer Discovery Protocol (LLDP) protocols, fielding configuration requests received on one of the X710-TM4's MC management interfaces such as NC-SI and handling special configuration requests received off an Ethernet port.

1.1.8.1 Protect, Detect and Recover

EMP firmware implements a design philosophy of platform resiliency with three attributes supporting the NIST Cyber Security Framework: protect, detect and recover, by verifying the firmware and critical device settings with built-in detection of corruption and automated device recovery to ensure returning the device to its originally programmed state. See [Section 3.4.9](#) and [Section 12.4](#) for details.

1.1.9 Internal Switch

The internal switch handles X710-TM4 packet buffering and also implements all its internal Ethernet switching capabilities. The internal switch can logically support up to 16 VLAN-aware bridges, connecting up to 384 VSIs out to X710-TM4's Ethernet ports via optional EVB S-components. The internal switch implements the filtering and forwarding behaviors expected by 802.1Q VLAN-aware bridges.



1.1.10 Receive Scheduler

The receive scheduler prioritizes received packets according to the ETS settings programmed at each Ethernet port.

1.1.11 Receive Filters

The X710-TM4 receive filters implement all of the X710-TM4's logic for queue selection. For each received packet, the forwarding process carried out by the internal switch selects which VSI the packet is associated with. The receive filters then determine which engine(s) handle the packet (EMP and LAN engine) and which selected VSI RQs the packet is associated with. Receive filters include flow director, RSS filters, etc.

1.1.12 Various Interfaces

1.1.12.1 Serial Flash Interface

The X710-TM4 provides an external Serial Peripheral Interface (SPI) serial interface to connect a Flash device. The X710-TM4 supports a 64 Mb (8 MB) serial Flash device.

The Flash device is required for storage of device firmware, device configuration parameters, identifiers that vary per adapter (like MAC addresses), and register overrides that auto load automatically after reset.

More information on the Serial Flash interface is available in [Section 3.4](#).

1.1.12.2 SMBus Interface

SMBus is an optional interface for PT and/or configuration traffic between an external MC and the X710-TM4. The X710-TM4's SMBus interface supports standard SMBus at 100 KHz and extensions up to a frequency of 1MHz. Refer to [Section 3.3.1](#) for an additional description of the SMBus interface, [Section 2.2.4](#) for the pin descriptions, [Section 9.5](#) for SMBus programming and [Section 13.6](#) for the timing characteristics.

1.1.12.3 NC-SI Interface

NC-SI is an optional interface for PT and/or configuration traffic between an external MC and the X710-TM4. Refer to [Section 3.3.2](#) for an additional description of the NC-SI interface, [Section 2.2.3](#) for the pin descriptions, [Section 9.6](#) for NC-SI programming, and [Section 13.6](#) for the timing characteristics.



1.1.12.4 Software Definable Pins (SDPs)

SDPs are typically used to exchange information with or to control external devices (such as PHY devices) under X710-TM4 software driver control. See [Section 3.5.2](#) for more details.

1.1.12.5 Light-emitting Diodes (LEDs)

The X710-TM4 implements eight output drivers intended for driving external LED circuits. The X710-TM4 can be configured so that either two of these outputs are allocated per port or four outputs are allocated per port (this later configuration is used when two ports are disabled). Each of the LED outputs can be individually configured to select which particular event, state, or activity it indicates. In addition, each LED can be individually configured for output polarity and for blinking versus non-blinking (steady-state) indications. See [Section 3.5.1](#) for more information.

1.2 Features

This section lists the X710-TM4's feature set.

Note: Refer to the *Intel® Ethernet Controller X710-TM4 Feature Support Matrix* for a list of features and interfaces supported by the X710-TM4.

Table 1-2. PCIe Host Interface Features

Description
PCIe v3.0 (8GT/s or 5GT/s or 2.5GT/s)
Number of lanes: x8, x4, x1
Supports optional PCIe link lane reversal
Requester Features <ul style="list-style-type: none"> • 64-bit address support for systems with more than 4 GB of physical memory • Maximum of 112 outstanding requests, allocated fairly as needed among PCI functions • Maximum payload size supported: 2 KB • Maximum read request size supported: 4 KB • All requests use TC TC0/VC0, the best effort service class for general purpose I/O • Optimized support for relaxed ordering transaction attribute • Optimized support for TLP Processing Hints (TPH) • Optimized support for ID-based Ordering (IDO)



Table 1-2. PCIe Host Interface Features

Description
<p>Completer Features</p> <ul style="list-style-type: none"> • Up to 16 Peripheral Component Interconnect (PCI) PFs using Alternative Routing-ID Interpretation (ARI) and up to 8 PFs using legacy routing-ID. The X710-TM4 can be configured to disable/hide any number of these PFs. The X710-TM4 is a Multi-function Per Port (MFP) device, meaning that it can support multiple PFs sharing a single Ethernet port. The X710-TM4 can also be configured with up to 8 PFs sharing one Ethernet port. • Each PF implements these Base Address Registers (BARs) <ul style="list-style-type: none"> – Memory BAR for direct access to most internal registers. This BAR can be configured to define either a 32-bit or 64-bit base address. It's size is 256 KB or larger. – MSI-X BAR for direct access to MSI-X-related structures. This BAR can be disabled/hidden. It can be configured to define either a 32-bit or 64-bit base address and it's size is 32 KB. – Expansion ROM BAR for direct host access to one of the X710-TM4's option ROM images. This BAR can be disabled/hidden. It defines a 32-bit base address and it's size is 64 KB or larger. – I/O BAR for indirect access to most internal registers. This BAR can be disabled/hidden. It defines a 32-bit base address and it's size is 32 bytes. • Per-PF capabilities list (some of these can be disabled/hidden) <ul style="list-style-type: none"> – PCI Power Management – MSI – MSI-X – PCIe – Vital Product Data (VPD) Each PF can access a single shared instance of VPD storage, with a size up to 1024 bytes – Advanced Error Reporting (AER) – Device Serial Number – Alternative RID Interpretation (ARI) – SR-IOV – TPH Requester – Access Control Services (ACS) – Secondary PCIe
<p>Reliability</p> <ul style="list-style-type: none"> • AER • Support for optional End-to-End CRC (ECRC) generation and checking • Recovery from data poisoning • Completion timeout
<p>Power Management</p> <ul style="list-style-type: none"> • Supports the PCI Power Management specification and section 5 of the PCI Express Base Specification • Active state power management (L0s and L1 states) • Does not support D1 or D2 power management states • Supports D0uninitialized, D0active, D3hot, and D3cold power management states • D3hot transition to D0 does preserve configuration context (as indicated by the RO PCI configuration bit No_Soft_Reset=1b)
<p>Interrupt Functionality</p> <ul style="list-style-type: none"> • INTx: Supports four INTx interrupts conveyed using the PCIe INTx virtual wire signaling mechanism. A given PF can only map a single INTx. • MSI: Supports one Message-Signaled Interrupt (MSI) per PF. • MSI-X: Supports up to 1168 MSI-X vectors, shared among PFs and VFs. Up to 129 MSI-X interrupts can be assigned to a single PF and up to 17 MSI-X interrupts can be assigned to a single VF, depending on the number of enabled functions. • Sophisticated interrupt moderation using Interrupt Throttling (ITR) and Interrupt Rate Limiting (INTRL).

**Table 1-2. PCIe Host Interface Features**

Description
<p>I/O Virtualization</p> <ul style="list-style-type: none"> • Supports PCI-SIG SR-IOV specification with up to 128 VFs. VFs can be flexibly assigned to PFs, although reassignment requires re-enumeration of the PCI bus hierarchy. • Each VF implements these BARs <ul style="list-style-type: none"> — Memory BAR for restricted access to Internal Registers. This BAR can be configured to define either a 32-bit or 64-bit base address. It's size is 16KB or larger. — MSI-X BAR for direct access to MSI-X-related structures. This BAR can be configured to define either a 32-bit or 64-bit base address. It's size is 16KB. • Per-VF Capabilities list (some of these can be disabled/hidden) <ul style="list-style-type: none"> — MSI-X — PCIe — AER — ARI — TPH requester — ACS

Table 1-3. Link Layer Ethernet Port Features

Description
Two integrated BASE-T ports capable of operating at 100/1G/2.5G/5G/10GBASE-T
Two serial technology ports capable of operating at 1 Gb/s or 10 Gb/s for SFP or backplane applications (SFI or KR). These two ports can also be used with an external BASE-T PHY
The X710-TM4's Maximum Transmit Unit Size (MTU) is 9728 - Ethernet header/CRC = 9728 - 18 = 9710 bytes (jumbo frames) MTU can be further reduced by additional header fields such as Virtual Local Area Network (VLAN) tag(s), etc.
Full-duplex operation at all supported speeds
Integrates support for IEEE Std 802.3 Clause 73 Auto-Negotiation for backplane Ethernet

Table 1-4. Performance on the Network vs. Packet Size

Description
<ul style="list-style-type: none"> • Maximize link capacity when operating at 10 Gb/s link with packets larger than 128 bytes at full duplex traffic • Maximize link capacity when operating at 2x 10 Gb/s links or 1x 10 Gb/s link in all packet sizes at full duplex traffic

Table 1-5. Transmit and Receive Scheduling

Description
<p>Queue Sets (see Section 1.1.4)</p> <ul style="list-style-type: none"> • Supports up to 1024 LAN queue sets (768 typically available) • Each queue set is assigned to the TC of a particular VSI • Round Robin (RR) bandwidth distribution between TQs assigned to same queue set • RR bandwidth distribution between LAN queue sets belonging to same TC of VSI



Table 1-5. Transmit and Receive Scheduling

Description
<p>Quanta:</p> <ul style="list-style-type: none"> • Transmit work is scheduled in units of configurable per-chip Quanta bytes • Quanta can be configured to be 1 KB, 2 KB, 4 KB, 8 KB, 16 B, 32 KB or 64 KB
<p>VSI:</p> <ul style="list-style-type: none"> • Supports up to 384 VSIs with an average of two TCs allocated per VSI • Independent ETS/SLA configuration per VSI • Configurable bandwidth limit per VSI
<p>VEB/VEPA:</p> <ul style="list-style-type: none"> • Supports up to 16 VEB/VEPA switching components • Configurable bandwidth allocation among VEB/VEPA VSIs • Configurable bandwidth Limit of VEB/VEPA egress port
<p>S-components:</p> <ul style="list-style-type: none"> • Supports up to 4 S-components, one per physical port • Configurable bandwidth allocation among S-channels • Independent ETS configuration per S-component egress port • Configurable bandwidth limit of S-component egress port
<p>Bandwidth Distribution:</p> <ul style="list-style-type: none"> • Supports two bandwidth allocation modes <ul style="list-style-type: none"> — Relative bandwidth allocation - used for ETS, and bandwidth allocation between ingress ports of the switching component — Best effort bandwidth allocation - used for SLA • Supports three arbitration schemes <ul style="list-style-type: none"> — Weighted Round Robin (RR) — Weighted Strict Priority (WSP) — Combination of WSP and weighted RR • On any given virtual link, minimum bandwidth allocation is 1% of the virtual link bandwidth
<p>Bandwidth Limit:</p> <ul style="list-style-type: none"> • Configurable bandwidth limit in range of Gb/s - 1 Mb/s with increment of 1 Mb/s • Configurable maximum bandwidth accumulation with a maximum of 200%

Table 1-6. Internal Switching Features

Description
<p>VSI Support</p> <p>X710-TM4 supports a total of 384 VSIs. VSI assignment is flexible, but the choice to support 384 VSIs is motivated by the following usage example:</p> <ul style="list-style-type: none"> • 256 VSIs for VFs or VMDq2 • 32 VSIs for PFs • 24 control ports (16 for VEBs/VEPAs, 4 for Ethernet ports, 4 for S-comp/E-comp) • 16 mirror ports • 4 for EMP • 20 extras

**Table 1-6. Internal Switching Features**

Description
<p>EVB and Bridge Port Extension (BPE) Functionality</p> <ul style="list-style-type: none"> • Supports EVB as defined in the IEEE P802.1Qbg specification • The X710-TM4 supports up to four S-components or four eBridge port extenders (one per port), up to 512 S-tags, and up to 384 S-channels • The X710-TM4 supports up to 16 internal VEBs or VEPAs • A single Ethernet port can connect up to 384 S-channels. Connecting to these S-channels can be VEBs, VEPAs, or port extenders. The X710-TM4 must be configured for either VEPA or port extenders (both are not supported simultaneously). Given these rules, plus the global constraints described in the preceding bullets, a single Ethernet port can connect either: <ul style="list-style-type: none"> — Up to 16 VEBs or VEPAs. This case only requires up to 16 S-channels. — Up to 384 EBPs or VEBs. Of the 384, only up to 16 can be VEBs. • Each VEB is compliant with the IEEE 802.1Q Ethernet VLAN-aware bridge specification • Supports MAC address forwarding table with 1536 entries • Supports MAC and VLAN address forwarding table with 2048 pairs • Supports VLAN insertion and removal for up to 512 VLAN tags located anywhere in the 4 K VLAN space • Supports private VLANs • Supports port mirroring • Each VSI can be programmed to receive promiscuous unicast, broadcast and multicast packets
<p>Overlay Network (cloud tunneling) supporting:</p> <ul style="list-style-type: none"> • VXLAN tunnel. Both parse and switching based on the network key • Geneve tunnel. Both parse and switching based on the network key • NVGRE tunnel. Both parse and switching based on the network key • IPinGRE tunnel. Both parse and switching based on the network key • MACinUDP tunnel. Both parse and switching based on the network key
<p>Internal switching operates independently of the state of the LAN ports (also when LAN ports are down)</p>

Table 1-7. System Manageability Features

Description
<p>Sideband interfaces to connect to an external MC</p> <ul style="list-style-type: none"> • SMBus operating at up to 1 Mb/s • NC-SI Interface at 100 Mb/s- DMTF standard DSP0222 • MCTP over SMBus/PCIe - DMTF standard DSP0261



Table 1-7. System Manageability Features

Description
<p>Sophisticated filters to select received packet flows for delivery or mirroring to the MC. Each of the following filters is instantiated per-Ethernet port:</p> <ul style="list-style-type: none"> • 4 MAC filters • 8 VLAN filters • 4 Ethertype filters • 4 to 7 IPv4/IPv6 filters • 16 UDP/TCP port filters • 1 Flexible Total Cost of Ownership (TCO) filter • Address Resolution Protocol (ARP) filtering • Neighbor discovery filtering • Remote Management Control Protocol (RMCP) filtering • Internet Control Message Protocol (ICMP) filtering
<p>Ability to internally switch packets for communication between an operating system and MC.</p>
<p>Statistics</p> <p>For each Ethernet port, the X710-TM4 implements the statistics defined in the following standards:</p> <ul style="list-style-type: none"> – IEEE Std 802.3 Clause 30 – RFC 2819 - RMON Ethernet statistics group <p>For each VSI, the X710-TM4 implements the statistics defined in the following standard: IEEE Std 802.3 Clause 30.</p>

Table 1-8. Power Management Features

Description
<p>Supports the PCIe power management features defined in the Power Management entry in Table 1-2.</p>
<p>Energy Efficient Ethernet (EEE)</p> <p>Supports EEE as defined in the IEEE P802.3az specification. EEE is supported with both internal backplane Ethernet PHYs and with select external PHYs. EEE is also supported on the internal BASE-T Ethernet PHY.</p> <ul style="list-style-type: none"> • Internal backplane Ethernet PHY modes that support EEE: KR and KX • Interfaces that support EEE using an external PHY: KR, SFI and SGMII • Internal BASE-T Ethernet PHY modes that support EEE: 10GBASE-TX and 10GBASE-T
<p>Low Power Link Up (LPLU)</p> <p>In a backplane Ethernet environment, when the X710-TM4 is entering a low power state like D3_{COLD}, the X710-TM4's LPLU logic attempts to re-autonegotiate its backplane Ethernet ports to their lowest possible link speed.</p>
<p>Advanced Power Management (APM) Wake Up</p> <p>Supports APM wake up per-PF:</p> <ul style="list-style-type: none"> • Magic packet filter. When a magic packet arrives, and if the Magic packet filter is enabled, this can trigger a wake up.

Table 1-9. General Features

Description
<p>Serial Flash Interface</p>
<p>Configurable LED operation for software or Original Equipment manufacturer (OEM) customization of LED displays. Default configuration by Non-volatile Memory (NVM) for all LEDs for pre-driver functionality</p>
<p>Device disable capability</p>

**Table 1-9. General Features**

Description
Package Size 22 x 22 mm
Watchdog timer
Time Sync (IEEE 1588)
Firmware and Expansion ROM Management <ul style="list-style-type: none"> All firmware and expansion ROM code is stored in flash memory and is field upgradeable Supports field upgrades via host software utility or BMC program Supports independent update of these objects: EMP firmware and Expansion ROM code Supports secure authentication of during an update procedure: EMP firmware expansion ROM code and protected with a digital signature produced using SHA256 hash and 2048b RSA encryption. A firmware and expansion ROM field upgrade procedure can be carried out while X710-TM4 is in normal operation. The X710-TM4 does not load and use the new content until a reset occurs. Failure of a firmware or expansion ROM field upgrade procedure (due to interrupting of Flash programming or authentication failure) never prevents a re-attempt of the field upgrade procedure. Normally X710-TM4 reverts to the previously saved firmware or expansion ROM content when a field upgrade fails.
SDPs <ul style="list-style-type: none"> 22 total SDPs: Four groups containing 4 pins, plus 6 groups containing a single pin. Each group can be configured for ownership by a different X710-TM4 software driver and can be independently programmed. SDPs are typically used to exchange information with or to control external devices (such as PHY devices) under X710-TM4 software driver control. Each SDP can be configured as an input, output, or interrupt source.
Device / Port / PCI Function Disable <p>The X710-TM4 implements two strapping pins that, together with NVM settings, enable the entire X710-TM4, or selected Ethernet ports and their associated PCI functions or selected PCI functions to be disabled. Disabling occurs before PCI enumeration, so the disabled resources are completely hidden.</p>

1.3 Conventions

1.3.1 Numbers and Number Bases

Hexadecimal numbers are written with a 0x prefix (like 0xFFFF or 0x1234ABDF). Binary numbers are written with a lowercase 'b' suffix (like 1001b or 10b). Decimal numbers are indicated without any suffix or using a lowercase 'd' suffix (like 4500d).

1.3.2 Byte Ordering

This section defines the internal organization of registers and memory structures that span multiple bytes. A few conventions to start with are:

1. Network Order — Ethernet always transmits multiple-bytes fields with the Most Significant (MS) byte first.
2. Endian notation — defines how a logical entity (such as a MAC address) is stored in a given structure (like register or descriptor). Two options exist:



- a. Little Endian (LE) notation — The MS byte of the logical entity is mapped to the highest byte address of the structure.
- b. Big Endian (BE) notation — The MS byte of the logical entity is mapped to the lowest byte address of the structure.

A few examples follow:

Example 1: A 32-bit counter is equal to 0x01234567 (such as the sequence number in the TCP header). The counter is transferred on the wire as: 01 23 45 67 where 01 is the first byte on the wire and 67 is the last byte.

LE registers store this counter as (in bytes) as follows:

0x01 - highest byte address

0x23

0x45

0x67 - lower byte address

BE registers store this counter as (in bytes) as follows:

0x67 - highest byte address

0x45

0x23

0x01 - lower byte address

Example 2: An L2 type register that holds the value of IPv4 header is equal to 0x0800. The field is transferred on the wire as: 08 00 where 08 is the first byte on the wire.

LE registers store this counter as (in bytes) as follows:

0x08 - highest byte address

0x00 - lower byte address

BE registers store this counter as (in bytes) as follows:

0x00 - highest byte address

0x08 - lower byte address

Example 3: A 48-bit Ethernet MAC address equals to 0x00112348A9BE. The Ethernet MAC address is transferred on the wire as: 00 11 23 48 A9 BE where 00 is the first byte on the wire.

LE registers store this counter as (in bytes) as follows:

0x00 - highest byte address

0x11

0x23

0x48

0xA9

0xBE - lower byte address

BE registers store this counter as (in bytes) as follows:

0xBE - highest byte address

0xA9



0x48

0x23

0x11

0x00 - lower byte address

The following rules determine the Endian-ness of the X710-TM4 structures:

- The general rule is that all structures are defined in LE notation unless defined otherwise. These structures include:
 - Registers
 - AQ commands
 - Structures in host memory (including any type of descriptors)
 - NVM
 - LAN
- The following structures are in BE notation:
 - Host memory buffers that are received or transmitted
 - Any structures that contains a MAC address (see exception for field vector)
- The following structures have a mixed notation:
 - Field vector is presented in mixed BE/LE notation: words are ordered in BE notation and bytes within the words are presented in LE notation.
 - Type-length-value (TLV) structures are stored in the NVM in mixed BE/LE notation: words are ordered in BE notation and bytes within the words are presented in LE notation.



1.4 Overview of Standards

Table 1-10 lists standards relevant to the X710-TM4.

Table 1-10. Standards Supported by the X710-TM4

Category	Description
Base	Title: Computing the Internet Checksum Document: http://www.ietf.org/rfc/rfc1071.txt Description: This RFC describes how to compute the Internet checksums used in IP, TCP and UDP.
Base	Title: A TCP/IP Tutorial Document: http://www.ietf.org/rfc/rfc1180.txt Description: Bare bones tutorial of TCP/IP protocol suite: ARP, IP and TCP and Upper Layer Protocols (ULPs). This is included for informative purposes only.
Base	Title: Implementing the Internet Checksum in Hardware. Document: http://www.ietf.org/rfc/rfc1936.txt Description: Techniques for efficiently implementing the Internet checksum in hardware.
Ethernet	Title: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications (IEEE Std 802.3-2012) Document: available from standards.ieee.org/getieee802 Description: Specifies the Ethernet MAC and PHY layers up to 100 Gb/s. Includes these now superseded docs (among many others): 802.3ae (10 Gb/s base spec), 802.3an (10GBASE-T), 802.3ap (backplane Ethernet, KX and KR)
Ethernet	Title: Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications – Amendment 8: MAC Control Frame for Priority-based Flow Control (IEEE P802.3bd-2011) Document: available from standards.ieee.org/getieee802 Description: Defines a MAC control frame to support 802.1Qbb priority-based flow control.
Ethernet	Title: IEEE Draft Standard for Ethernet Amendment 2: Physical Layer Specifications and Management Parameters for 100 Gb/s Operation Over Backplanes and Copper Cable (IEEE P802.3bj/D3.1) Document: available from standards.ieee.org/getieee802
Ethernet	Title: IEEE Standard for Local and Metropolitan Area Networks – Media access control (MAC) Bridges (IEEE Std 802.1D-2004) Document: available from standards.ieee.org/getieee802 Description: Base specification for Ethernet bridging.
Ethernet	Title: “IEEE Standards for Local and Metropolitan Area Networks - Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks” (IEEE Std 802.1Q-2011) Document: available from standards.ieee.org/getieee802 Description: Ethernet VLAN-aware bridge specification.
Ethernet	Title: IEEE Standard for Local and metropolitan area networks— Link Aggregation (IEEE Std 802.1AX-2008) Document: available from standards.ieee.org/getieee802 Description: Logic and protocols that enable aggregation of one or more Ethernet links into a single logical link. Until recently, link aggregation was defined in the 802.3 specification, but in the 2008 version it was moved to 802.1.

**Table 1-10. Standards Supported by the X710-TM4 (Continued)**

Category	Description
Ethernet	<p>Title: Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks — Amendment 17: Priority-based Flow Control (IEEE P802.1Qbb-2011)</p> <p>Document: available from standards.ieee.org/getieee802</p> <p>Description: Priority-based Flow Control (PFC) is one of the specifications that comprise DCB. PFC enables flow control per TC on IEEE 802 point-to-point full duplex links. This is achieved by a mechanism similar to the IEEE Std. 802.3-2018 Annex 31B PAUSE, but operating on individual priorities.</p>
Ethernet	<p>Title: Virtual Bridged Local Area Networks — Amendment 18: Enhanced Transmission Selection for Bandwidth Sharing Between Traffic Classes (IEEE P802.1Qaz-2011)</p> <p>Document: available from standards.ieee.org/getieee802</p> <p>Description: ETS is one of the specifications that comprise DCB. ETS enables arbitration of bandwidth between TCs. This specification also defines Data Center Bridging Exchange (DCBX) protocol. DCBX enables configuration of DCB features onto an Ethernet LAN.</p>
Ethernet	<p>Title: Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks — Amendment 21: Edge Virtual Bridging (IEEE P802.1Qbg-2012)</p> <p>Document: available from standards.ieee.org/getieee802</p> <p>Description: EVB is one of the specifications that comprise DCB. EVB defines many of the virtual switching features on end stations like the X710-TM4. This includes definition of things like S-channels, which enable the multiplexing of multiple virtual channels on a single physical LAN, VSIs, VEBs, VEPAs, etc. It also defines new management infrastructure for administering the new features.</p>
DCB	<p>Title: DCB Capability Exchange Protocol Base Specification, Rev 1.01</p> <p>Document: available from http://www.ieee802.org/1/files/public/docs2008/az-wadekar-dcbx-capability-exchange-discovery-protocol-1108-v1.01.pdf</p> <p>Description: Defines CEE DCBX, a pre-standard version of the DCB Capability Exchange Protocol</p>
DCB	<p>Title: Priority Grouping for DCB Networks, Rev 1.01</p> <p>Document: available from http://www.ieee802.org/1/files/public/docs2008/az-wadekar-ets-proposal-0608-v1.01.pdf</p> <p>Description: Defines CEE ETS, a pre-standard version of the DCB Enhanced Transmission Selection Protocol</p>
Ethernet	<p>Title: Virtual Bridged Local Area Networks — Bridge Port Extension (IEEE P802.1BR/D3.3)</p> <p>Document: available from standards.ieee.org/getieee802</p> <p>Description: BPE is a dedicated specification describing the bridge port extender element. It specifies the use of E-channels and a multicast replication service to extend bridge ports across multiple physical or virtual devices.</p>
Ethernet	<p>Title: IEEE Standard for Local and metropolitan area networks— Station and Media Access Control Connectivity Discovery (IEEE Std 802.1AB-2009)</p> <p>Document: available from standards.ieee.org/getieee802</p> <p>Description: Defines Link Layer Discovery Protocol (LLDP) that enables a server to advertise its identity, capabilities, and interconnections to other entities on an Ethernet fabric.</p>
Ethernet	<p>Title: IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems (IEEE Std 1588-2008)</p> <p>Document: available from standards.ieee.org</p> <p>Description: Defines a protocol that enables precise synchronization of clocks in systems communicating via packet networks.</p>



Table 1-10. Standards Supported by the X710-TM4 (Continued)

Category	Description
Ethernet	<p>Title: SFF-8431 Specifications for Enhanced Small Form Factor Pluggable Module SFP+, rev 4.1, 7/6/2009</p> <p>Document: ftp://ftp.seagate.com/sff/SFF-8431.PDF (or see www.sffcommittee.com)</p> <p>Description: Defines the SerDes Framer Interface (SFI) high speed electrical interface to a Small Form-factor Pluggable (SFP+) optical module. Also defines the SFP+ management interface.</p>
Ethernet	<p>Title: RMI Specification, rev 1.2, 3/20/1998</p> <p>Description: Reduced pin count interface used in place of IEEE standard Media Independent Interface (MII). The X710-TM4 has an Reduced Media Independent Interface (RMII) interface for its NC-SI connection.</p>
Ethernet	<p>Title: Serial-GMII Specification, rev 1.8, 11/2/2005, published by Cisco Systems</p> <p>Document: ftp://ftp-eng.cisco.com/smii/sgmii.pdf</p> <p>Description: Reduced pin count interface used in place of IEEE standard Gigabit Media Independent Interface (GMII).</p>
Ethernet	<p>Title: Ethernet Alliance, Ethernet Jumbo Frames, ver 0.3, 11/17/2009</p> <p>Document: EA-Ethernet Jumbo Frames v0 3.pdf</p> <p>Title: Extended Frame Sizes for Next Generation Ethernet</p> <p>Document: AlteonExtendedFrames_W0601.pdf</p> <p>Title: Extended Ethernet Frame Size Support</p> <p>Document: draft-kaplan-isis-ext-eth-02.txt</p> <p>Description: Documents describing jumbo frames. Included for informative purposes only.</p>
Test	<p>Title: IEEE Standard Test Access Port and Boundary-Scan Architecture (IEEE Std 1149.1-2001)</p> <p>Document: available from standards.ieee.org</p> <p>Description: Defines a standard interface through which instructions and test data are communicated to an integrated circuit for component- and circuit board-level testing.</p>
Ethernet/ IP	<p>Title: Standard for the Transmission of IP Datagrams over Ethernet Networks</p> <p>Document: http://www.ietf.org/rfc/rfc894.txt</p> <p>Description: This specifies the method for transmitting IP datagrams over Ethernet.</p>
Ethernet/ IP	<p>Title: Standard for the Transmission of IP Datagrams over IEEE 802 Networks</p> <p>Document: http://www.ietf.org/rfc/rfc1042.txt</p> <p>Description: This specifies the method for transmitting IP datagrams over IEEE Std. 802.3-2018 networks.</p>
iARP	<p>Title: Address Resolution Protocol (ARP)</p> <p>Document: http://www.ietf.org/rfc/rfc0826.txt</p> <p>Description: Protocol to convert IP addresses to Ethernet addresses</p>
IP	<p>Title: Internet Protocol</p> <p>Document: http://www.ietf.org/rfc/rfc0791.txt</p> <p>Description: Base specification for IPv4.</p>
IP	<p>Title: Internet Protocol, Version 6</p> <p>Document: http://www.ietf.org/rfc/rfc2460.txt</p> <p>Description: Base specification for IPv6.</p>

**Table 1-10. Standards Supported by the X710-TM4 (Continued)**

Category	Description
IP	Title: Neighbor Discovery for IP version 6 (IPv6) Document: http://www.ietf.org/rfc/rfc4861.txt Description: IPv6 nodes use neighbor discovery to discover each other's presence, to determine each other's link-layer addresses, to find routers, and to maintain reachability information. This is an important standard for power management.
IP	Title: Multicast Listener Discovery (MLD) for IPv6 Document: http://www.ietf.org/rfc/rfc2710.txt Description: Specifies the protocol used by an IPv6 router to discover the multicast listeners on its directly attached links. MLD is derived from version 2 of IPv4's Internet Group Management Protocol, IGMPv2. This is an important standard for power management.
IP	Title: Multicast Listener Discovery Version 2 (MLDv2) for IPv6 Document: http://www.ietf.org/rfc/rfc3810.txt Description: Updates RFC 2710. This is an important standard for power management.
IP	Title: TCP Processing of the IPv4 Precedence Field Document: http://www.ietf.org/rfc/rfc2873.txt Description: Corrects a conflict between TCP as defined in RFC793 and DiffServ in handling of the IPv4 precedence field.
TCP	Title: Transmission Control Protocol Document: http://www.ietf.org/rfc/rfc0793.txt Description: Base specification for TCP.
TCP	Title: Window and Acknowledgment Strategy in TCP Document: http://www.ietf.org/rfc/rfc813.txt
TCP	Title: Congestion Control in IP/TCP Internetworks Document: http://www.ietf.org/rfc/rfc896.txt Description: Defines the Nagle algorithm, which specifies delaying transmission of small amounts of data when there are Acknowledgments (ACKs) outstanding.
TCP	Title: TCP Extensions for High Performance Document: http://www.ietf.org/rfc/rfc1323.txt Description: Defines the TCP window scale and timestamp options, Round Trip Time Measurement (RTTM) and Protect Against Wrapped Sequences (PAWS).
TCP	Title: Known TCP Implementation Problems Document: http://www.ietf.org/rfc/rfc2525.txt Description: This RFC describes implementation issues with various historical TCP/IP stacks. While it is included here for informative purposes only, it does serve as a good check list to avoid known problems.
TCP	Title: TCP Congestion Control Document: http://www.ietf.org/rfc/rfc5681.txt Description: TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms.
TCP	Title: The NewReno Modification to TCP's Fast Recovery Algorithm Document: http://www.ietf.org/rfc/rfc3782.txt Description: Update the fast recovery algorithm that is run after three duplicate ACKs have been received. Changes to the CWND during fast recovery, scheduling additional fast retransmitted packets if the received ACKs do not acknowledge all the data when fast recovery was entered.



Table 1-10. Standards Supported by the X710-TM4 (Continued)

Category	Description
TCP	<p>Title: TCP Problems with Path MTU Discovery Document: http://www.ietf.org/rfc/rfc2923.txt Description: Discusses problems with existing RFC 1191 implementations that should be avoided. Serves as an implementation checklist.</p>
TCP	<p>Title: Computing TCP's Retransmission Timer Document: http://www.ietf.org/rfc/rfc2988.txt Description: Defines the standard algorithm that TCP senders are required to use to compute and manage their retransmission timer.</p>
TCP	<p>Title: Increasing TCP's Initial Window Document: http://www.ietf.org/rfc/rfc3390.txt Description: Specifies an optional standard for TCP to increase the permitted initial window from one or two segments to roughly 4KB.</p>
TCP	<p>Title: TCP Congestion Control with Appropriate Byte Counting (ABC) Document: http://www.ietf.org/rfc/rfc3465.txt Description: This experimental RFC defines a small modification to the way TCP increases its congestion window. Instead of increasing CWND by a constant amount for each acknowledgment, CWND is increased based on the number of previously unacknowledged bytes each ACK covers.</p>
UDP	<p>Title: User Datagram Protocol Document: http://www.ietf.org/rfc/rfc0768.txt Description: Base specification for UDP.</p>
Tunneling	<p>Title: NVGRE: Network Virtualization using Generic Routing Encapsulation Document: http://datatracker.ietf.org/doc/draft-sridharan-virtualization-nvgre/?include_text=1 Description: MAC in Generic Routing Encapsulation (GRE) over IP encapsulation</p>
Tunneling	<p>Title: VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks Document: http://datatracker.ietf.org/doc/draft-mahalingam-dutt-dcops-vxlan/?include_text=1 Description: MAC in UDP encapsulation</p>
Mgmt	<p>Title: System Management Bus (SMBus) Specification, v3.0, 12/20/2014" Document: http://www.smbus.org/specs/SMBus_3_0_20141220.pdf Description: A two-wire interface for communication of management information, based on the principles of operation of I²C.</p>
Mgmt	<p>Title: Network Controller Sideband Interface (NC-SI) Spec, v1.0.1, 1/10/2013 Document: dmtf.org/sites/default/files/standards/documents/DSP0222_1.0.pdf Description: Standardizes the sideband communication interface between a NIC (the X710-TM4) and a MC.</p>
Mgmt	<p>Title: Network Controller Sideband Interface (NC-SI) Spec, v1.1.0 Draft 40 11/08/2015 Document: https://sharepoint.ger.ith.intel.com/sites/LAD_SIE_GP1/Architecture/Standards/Shared%20Documents/Manageability/NC-SI/DSP0222_1.1.0_Draft40.doc Description: Standardizes the sideband communication interface between a NIC (the X710-TM4) and a MC.</p>
Mgmt	<p>Title: UM10204 I²C-bus specification and user manual rev 5. Document: www.nxp.com/documents/user_manual/UM10204.pdf Description: I²C spec with support up to 1 Mbps</p>

**Table 1-10. Standards Supported by the X710-TM4 (Continued)**

Category	Description
Mgmt	Title: Management Component Transport Protocol (MCTP) Base Specification, rev 1.1.0, 4/22/2010 Document: dmtf.org/sites/default/files/standards/documents/DSP0236_1.1.0.pdf Description: Specifies MCTP protocol.
Mgmt	Title: Management Component Transport Protocol (MCTP) SMBus/I ² C Transport Binding Specification, rev 1.0.0, 7/28/2009 Document: dmtf.org/sites/default/files/standards/documents/DSP0237_1.0.0.pdf Description: Describes the binding of MCTP over SMBus.
Mgmt	Title: Management Component Transport Protocol (MCTP) PCIe VDM Transport Binding Specification, rev 1.0.1, 12/11/2009 Document: dmtf.org/sites/default/files/standards/documents/DSP0238_1.0.1.pdf Description: Describes the binding of MCTP over PCIe.
Mgmt	Title: Management Component Transport Protocol (MCTP) IDs and Codes, rev 1.1.0, 11/3/2009 Document: dmtf.org/sites/default/files/standards/documents/DSP0239_1.1.0.pdf Description: Describes constants used by MCTP specifications.
Mgmt	Title: NC-SI Over MCTP Specification, rev 1.1.0, 3/21/2015 Document: http://www.dmtf.org/sites/default/files/standards/documents/DSP0261_1.1.0.pdf Description: Describes the encapsulation of NC-SI packets in MCTP.
Power Mgmt	Title: Magic Packet Technology, November 1995 Document: http://support.amd.com/TechDocs/20213.pdf Description: Defines a method for waking up a sleeping networked PC using a specific Ethernet frame (a Magic packet).
Power Mgmt	Title: PCI Bus Power Management Interface Specification, rev 1.2, 3/3/2004 Document: available from www.pcisig.com Description: Defines a standard set of PCI peripheral power management hardware interfaces and behavioral policies.
Power Mgmt	Title: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications – Amendment: Media Access Control parameters, Physical Layers and management parameters for Energy-Efficient Ethernet (IEEE P802.3az-2010) Document: available from standards.ieee.org/getieee802
Power Mgmt	Title: proxZZZy for sleeping hosts, February 2010 (ECMA-393) Document: www.ecma-international.org/publications/files/ECMA-ST/ECMA-393.pdf Description: Defines a low-power proxy that handles key network tasks for a high-power device, thus allowing the high-power device to sleep when not in active use.
Statistics	Title: The Interfaces Group MIB Document: http://www.ietf.org/rfc/rfc2863.txt Description: Describes objects used for managing network interfaces.
Statistics	Title: Remote Network Monitoring MIB Document: http://www.ietf.org/rfc/rfc2819.txt Description: Defines objects for managing remote network monitoring devices. Includes the popular packet size histogram counters.
Statistics	Title: Microsoft NDIS 6.0 OID_GEN_STATISTICS Document: msdn.microsoft.com/en-us/library/ff569640(VS.85).aspx Description: Adapter statistics counters defined by Microsoft for NDIS 6.0.



Table 1-10. Standards Supported by the X710-TM4 (Continued)

Category	Description
Statistics	Title: MIB for the Internet Protocol (IP) Document: http://www.ietf.org/rfc/rfc4293.txt Description: IP version-independent IP MIB. Obsoletes RFC 2011, 2465, 2466.
Statistics	Title: MIB for the Transmission Control Protocol (TCP) Document: http://www.ietf.org/rfc/rfc4022.txt Description: IP version-independent TCP MIB. Obsoletes RFC 2012, 2452.
Statistics	Title: MIB for the User Datagram Protocol (UDP) Document: http://www.ietf.org/rfc/rfc4113.txt Description: Objects for managing implementations of UDP. Obsoletes RFC 2013.
PCI	Title: PCI Local Bus Specification, rev 3.0, 2/3/2004 Document: available from www.pcisig.com Description: Compliant with select sections, such as Appendix I Vital Product Data.
PCI	Title: PCI Hot-Plug Specification, rev 1.1, 6/20/2001 Document: available from www.pcisig.com Description: Defines how PCI add-in cards are installed and removed while the system is running.
PCI	Title: PCI Firmware Specification, rev 3.0, 6/20/2005 Document: available from www.pcisig.com Description: Defines the firmware interface for managing PCIe systems in a host computer. Describes the format, contents, and code entry points for expansion ROMs.
PCI	Title: PCI Express Base Specification, rev 3.0, 11/10/2010 Document: available from www.pcisig.com Description: Contains the technical details of the PCIe architecture, protocol, link layer, physical layer, and software interface. Also defines some important power management features, including Optimized Buffer Flush/Fill (OBFF).
PCI	Title: PCI Express Card Electromechanical Specification, rev 2.0, 4/11/2007 Document: available from www.pcisig.com Description: Mechanical and electrical specifications for an Evo card form factor.
PCI-IOV	Title: Single Root I/O Virtualization and Sharing Specification, rev 1.1, 1/20/2010 Document: available from www.pcisig.com Description: The SR-IOV specification defines extensions to the PCIe specification that enable the VMs in a virtualized server to efficiently share PCI adapter hardware resources.
Software	Title: Microsoft specification for "Receive Side Scaling" (RSS) from MSDN. Document: msdn.microsoft.com/en-us/library/ff567236%28v=VS.85%29.aspx Description: RSS is a network driver technology that enables the efficient distribution of network receive processing across multiple CPUs in multiprocessor systems.



2.0 Pin Interface

2.1 Pin Descriptions

This section provides detailed descriptions of the X710-TM4 signal pins, grouped by function. A “_N” following the signal name indicates that the signal is active-low. Signal names with a suffix of “_p” and “_n” refer to differential signals.

The buffer types are listed in [Table 2.1](#).

Table 2-1. Buffer Types

Buffer	Description
In	Input is a standard input-only signal.
Out (O)	Totem Pole Output (TPO) is a standard active driver.
t/s	Tri-state is a bi-directional, tri-state input/output pin.
o/d	Open drain enables multiple devices to share as a wire-OR.
A-in	Analog input signals.
A-out	Analog output signals.
A-Inout	Bi-directional analog signals.
B	Input BIAS.
CML-in	Current Mode Logic (CML) input signal.
NCSI-in	NC-SI input signal.
NCSI-out	NC-SI output signal.
Pu	Internal pull-up resistor.
Pd	Internal pull-down resistor.

2.2 Pin Assignment and Description

The X710-TM4 is packaged in a 22 mm x 22 mm 503-pin Flip-Chip Ball Grid Array (FCBGA) package. The following sections provide the signal names, pin/ball assignments and signal descriptions. The electrical specifications for the signals are described in [Section 13.0](#).



2.2.1 PCIe Interface Pins

This section provides the pin assignment for PCIe interface signals. The AC/DC specifications for the PCIe interface signals are defined in [Section 13.6](#).

Table 2-2. PCIe Interface Signals

Signal	Ball #	Type	Description
PE_CLK_p PE_CLK_n	AA1 AB2	A-in	PCIe Differential Reference Clock In. A 100 MHz differential clock input. This clock is used as the reference clock for the PCIe Tx/Rx circuitry and by the PCIe core PLL to generate clocks for the PCIe core logic.
PET_0_p PET_0_n	AD4 AC5	A-out	PCIe Serial Data Output. A serial differential output pair running at 8 Gb/s or 5 Gb/s or 2.5 Gb/s. This output carries both data and an embedded 8 GHz or 5 GHz or 2.5 GHz clock that is recovered along with data at the receiving end.
PET_1_p PET_1_n	AD8 AD10	A-out	Same as previous.
PET_2_p PET_2_n	AD14 AD16	A-out	Same as previous.
PET_3_p PET_3_n	AD20 AD22	A-out	Same as previous.
PET_4_p PET_4_n	AD26 AD28	A-out	Same as previous.
PET_5_p PET_5_n	AC31 AD32	A-out	Same as previous.
PET_6_p PET_6_n	AC35 AD36	A-out	Same as previous.
PET_7_p PET_7_n	AC39 AD40	A-out	Same as previous.
PER_0_p PER_0_n	AA5 AA7	A-in	PCIe Serial Data Input. A serial differential input pair running at 8 Gb/s or 5 Gb/s or 2.5 Gb/s. An embedded clock present in this input is recovered along with the data.
PER_1_p PER_1_n	AB10 AB12	A-in	Same as previous.
PER_2_p PER_2_n	AB16 AB18	A-in	Same as previous.
PER_3_p PER_3_n	AB22 AB24	A-in	Same as previous.
PER_4_p PER_4_n	AA27 AB28	A-in	Same as previous.
PER_5_p PER_5_n	Y30 AA31	A-in	Same as previous.
PER_6_p PER_6_n	Y34 AA35	A-in	Same as previous.



Signal	Ball #	Type	Description
PER_7_p PER_7_n	Y38 AA39	A-in	Same as previous.
PE_WAKE_N	P2	o/d	Wake. Pulled to 0b to indicate that a Power Management Event (PME) is pending and the PCIe link should be restored. Defined in the PCIe specifications.
PE_RST_N	R1	In	Power and Clock Good Indication. Indicates that power and PCIe reference clock are within specified values. Defined in the PCIe specifications. Also called PCIe reset and PERST.

2.2.2 Ethernet Interface Pins

This section provides the pin assignments for Ethernet interface signals. The AC/DC specifications for the Ethernet interface signals are defined in [Section 13.6](#).

Note: Refer to [Table 2-14](#) for specific pins that apply to the X710-AT2.

Table 2-3. Ethernet Interface Pins

Signal	Ball #	Type	Description
XTAL_P XTAL_N	D42 E41	A-in	10GBASE-T PHY Reference Clock Inputs. These pins accept the clock based on the configuration settings of the XTAL_BYPASS and REFCLK_SEL pins. A continuous 50 MHz (or 156.25 MHz) reference clock must be supplied to the PHY. XTAL_N and XTAL_P make up the reference clock for ports 0 and 1. When the XTAL_BP pin is low, apply a 50 MHz crystal oscillator only. When crystal bypass is high, these pins are configured to accept a differential reference clock. In this mode, the XTAL_P/N should be AC-coupled to the system clock.
REFCLKIN_p_XTAL_IN REFCLKIN_n_XTAL_OUT	C1 C3	CML-in	External Reference Clock Input or Crystal Oscillator Input/Output. These pins might be driven by a 25 MHz crystal or an external clock oscillator (25 MHz $\pm 0.01\%$). The OSC_SEL pin is used to choose between a crystal source or an external clock oscillator.
SER0_RX_n SER0_RX_p	A13 B14	A-in	Serial Data Input for Ethernet interface. A serial differential input pair running at up to 10.3125 GBaud. An embedded clock present in this input is recovered along with the data.
SER0_TX_n SER0_TX_p	B18 A17	A-out	Serial Data Output for Ethernet Interface. A serial differential output pair running at up to 10.3125 GBaud. This output carries both data and an embedded clock that is recovered along with data at the receiving end.
SER1_RX_n SER1_RX_p	A5 B6	A-in	Serial Data Input for Ethernet interface. A serial differential input pair running at up to 10.3125 GBaud. An embedded clock present in this input is recovered along with the data.
SER1_TX_n SER1_TX_p	B10 A9	A-out	Serial Data Output for Ethernet Interface. A serial differential output pair running at up to 10.3125 GBaud. This output carries both data and an embedded clock that is recovered along with data at the receiving end.



Table 2-4. External Ethernet PHY Control - MDIO / I²C Interface Signals

Signal	Ball #	Type	Description
MDIO0_SDA0	N1	T/s, o/d	Management Data, when configured as an MDIO interface. Bi-directional signal for serial data transfers between the X710-TM4 and the PHY management registers for port 0. Note: Tri-state buffer, requires an external pull-up device. I ² C Data, when configured as 2-wire management interface for port 0. Stable during the high period of the clock (unless it is a start or stop condition). Note: Open drain buffer requires an external pull-up device.
MDC0_SCL0	N3	O, o/d	Management Clock, when configured as an MDIO interface. Clock output for accessing the PHY management registers for port 0. MDC clock frequency is proportional to link speed. At 10 Gb/s Link speed, MDC frequency can be set to 2.4 MHz (default) or 24 MHz. I ² C Clock, when configured as 2-wire management interface for port 0. One clock pulse is generated for each data bit transferred. Note: This I/O operates as an open drain buffer, and therefore requires an external pull-up device.
MDIO1_SDA1	M2	T/s, o/d	Management Data, when configured as an MDIO interface. Bi-directional signal for serial data transfers between the X710-TM4 and the PHY management registers for port 1. Note: Tri-state buffer requires an external pull-up device. I ² C Data, when configured as 2-wire management interface for port 1. Stable during the high period of the clock (unless it is a start or stop condition). Note: Open drain buffer requires an external pull-up device.
MDC1_SCL1	M4	O, o/d	Management Clock, when configured as an MDIO interface. Clock output for accessing the PHY management registers for port 1. MDC clock frequency is proportional to link speed. At 10 Gb/s Link speed, MDC frequency can be set to 2.4 MHz (default) or 24 MHz. I ² C Clock, when configured as 2-wire management interface for port 1. One clock pulse is generated for each data bit transferred. Note: This I/O operates as an open drain buffer, and therefore requires an external pull-up device.
MDIO2_SDA2	L1	T/s, o/d	Management Data, when configured as an MDIO interface. Bi-directional signal for serial data transfers between the X710-TM4 and the PHY management registers for port 2. Note: Tri-state buffer, requires an external pull-up device. I ² C Data, when configured as 2-wire management interface for port 2. Stable during the high period of the clock (unless it is a start or stop condition). Note: Open drain buffer requires an external pull-up device.
MDC2_SCL2	L3	O, o/d	Management Clock, when configured as an MDIO interface. Clock output for accessing the PHY management registers for port 2. MDC clock frequency is proportional to link speed. At 10 Gb/s Link speed, MDC frequency can be set to 2.4 MHz (default) or 24 MHz. I ² C Clock, when configured as 2-wire management interface for port 2. One clock pulse is generated for each data bit transferred. Note: This I/O operates as an open drain buffer, and therefore requires an external pull-up device.
MDIO3_SDA3	K2	T/s, o/d	Management Data, when configured as an MDIO interface. Bi-directional signal for serial data transfers between the X710-TM4 and the PHY management registers for port 3. Note: Tri-state buffer requires an external pull-up device. I ² C Data, when configured as 2-wire management interface for port 3. Stable during the high period of the clock (unless it is a start or stop condition). Note: Open drain buffer requires an external pull-up device.
MDC3_SCL3	K4	O, o/d	Management Clock, when configured as an MDIO interface. Clock output for accessing the PHY management registers for port 3. MDC clock frequency is proportional to link speed. At 10 Gb/s Link speed, MDC frequency can be set to 2.4 MHz (default) or 24 MHz. I ² C Clock, when configured as 2-wire management interface for port 3. One clock pulse is generated for each data bit transferred. Note: This I/O operates as an open drain buffer, and therefore requires an external pull-up device.



Signal	Ball #	Type	Description
MDC	L41	Pu	Management Data Clock Input for 10GBASE-T. Connects through a level shifter to MDC_0_SCL0. The MDC clock input must be provided to enable MII management functions. The MDIO signal is 3.3V tolerant when it is configured for a 2.5V signal.
MDIO	M42	Pu	Management Data I/O for 10GBASE-T. Connects through a level shifter to MDIO_0_SDA0. The serial input/output bit is used to read from and write to the MII registers. The data value on the MDIO pin is valid and latched on the rising edge of MDC. The MDIO signal is 3.3V tolerant when it is configured for a 2.5V signal.
P0_A_P P0_A_N	B40 A39	A-in A-out	Port 0 Transmit and Receive MDI Pair. Differential data from copper media is transmitted and received on four signal pairs for 10GBASE-T, 5GBASE-T, 2.5GBASE-T and 1000BASE-T modes. In auto-negotiation and 100BASE-TX mode, data is transmitted by the BASE-T PHY on P0/1_A_P/N and received on P0/1_B_P/N.
P0_B_P P0_B_N	A37 B38	A-in A-out	Port 0 Transmit and Receive MDI Pair. Differential data from copper media is transmitted and received on four signal pairs for 10GBASE-T, 5GBASE-T, 2.5GBASE-T and 1000BASE-T modes. In auto-negotiation and 100BASE-TX mode, data is transmitted by the BASE-T PHY on P0/1_A_P/N and received on P0/1_B_P/N.
P0_C_P P0_C_N	B36 A35	A-in A-out	Port 0 Transmit and Receive MDI Pair. Differential data from copper media is transmitted and received on four signal pairs for 10GBASE-T, 5GBASE-T, 2.5GBASE-T and 1000BASE-T modes. In auto-negotiation and 100BASE-TX mode, data is transmitted by the BASE-T PHY on P0/1_A_P/N and received on P0/1_B_P/N.
P0_D_P P0_D_N	A33 B34	A-in A-out	Port 0 Transmit and Receive MDI Pair. Differential data from copper media is transmitted and received on four signal pairs for 10GBASE-T, 5GBASE-T, 2.5GBASE-T and 1000BASE-T modes. In auto-negotiation and 100BASE-TX mode, data is transmitted by the BASE-T PHY on P0/1_A_P/N and received on P0/1_B_P/N.
P1_A_P P1_A_N	A23 B24	A-in A-out	Port 1 Transmit and Receive MDI Pair. Differential data from copper media is transmitted and received on four signal pairs for 10GBASE-T, 5GBASE-T, 2.5GBASE-T and 1000BASE-T modes. In auto-negotiation and 100BASE-TX mode, data is transmitted by the BASE-T PHY on P0/1_A_P/N and received on P0/1_B_P/N.
P1_B_P P1_B_N	B26 A25	A-in A-out	Port 1 Transmit and Receive MDI Pair. Differential data from copper media is transmitted and received on four signal pairs for 10GBASE-T, 5GBASE-T, 2.5GBASE-T and 1000BASE-T modes. In auto-negotiation and 100BASE-TX mode, data is transmitted by the BASE-T PHY on P0/1_A_P/N and received on P0/1_B_P/N.
P1_C_P P1_C_N	A27 B28	A-in A-out	Port 1 Transmit and Receive MDI Pair. Differential data from copper media is transmitted and received on four signal pairs for 10GBASE-T, 5GBASE-T, 2.5GBASE-T and 1000BASE-T modes. In auto-negotiation, and 100BASE-TX mode, data is transmitted by the BASE-T PHY on P0/1_A_P/N and received on P0/1_B_P/N.
P1_D_P P1_D_N	B30 A29	A-in A-out	Port 1 Transmit and Receive MDI Pair. Differential data from copper media is transmitted and received on four signal pairs for 10GBASE-T, 5GBASE-T, 2.5GBASE-T and 1000BASE-T modes. In auto-negotiation, and 100BASE-TX mode, data is transmitted by the BASE-T PHY on P0/1_A_P/N and received on P0/1_B_P/N.

Note: If the I²C is disconnected, an external pull-up should be used for the clock and data pins.

2.2.3 NC-SI Interface Pins

This section provides the pin assignment for NC-SI signals. The AC/DC specifications for the NC-SI interface signals are defined in [Section 13.6](#).

**Table 2-5. NC-SI Interface Signals**

Signal	Ball #	Type	Description
NCSI_CLK_IN	H2	NCSI-In	NC-SI Reference Clock Input. Synchronous clock reference for receive, transmit, and control interface. It is a 50 MHz clock \pm 100 ppm.
NCSI_CRS_DV	J5	NCSI-Out	Carrier Sense/Receive Data Valid (CRS/DV).
NCSI_RXD_0 NCSI_RXD_1	J1 J3	NCSI-Out	Receive Data. Data signals to the MC.
NCSI_TX_EN	H4	NCSI-In	Transmit Enable.
NCSI_TXD_0 NCSI_TXD_1	F2 G1	NCSI-In	Transmit Data. Data signals from the MC.
NCSI_ARB_IN	G3	NCSI-In	NC-SI Hardware Arbitration Input. If GL_MNG_HWARB_CTRL.NCSI_ARB_EN is cleared, this pin is internally pulled up.
NCSI_ARB_OUT	F4	NCSI-Out	NC-SI Hardware Arbitration Output.

2.2.4 SMBus Interface Pins

This section provides the pin assignment for the SMBus interface signals. The AC/DC specifications for the SMBus interface signals are defined in [Section 13.6](#).

Table 2-6. SMBus Interface Signals

Signal	Ball #	Type	Description
SMBCLK	R39	o/d	SMBus Clock. One clock pulse is generated for each data bit transferred. 3.3V tolerant.
SMBD	T40	o/d	SMBus Data. Stable during the high period of the clock (unless it is a start or stop condition). 3.3V tolerant.
SMBALRT_N	U41	o/d	SMBus Alert. Acts as an interrupt pin of a slave device on the SMBus. 3.3V tolerant.

Note: If the SMBus is disconnected, an external pull-up should be used for the SMBALRT_N, SMBCLK and SMBD pins.

2.2.5 Serial Flash Memory Interface Pins

This section provides the pin assignment for SPI signals for connectivity to Flash memory devices. The AC/DC specifications for the serial Flash memory interface signals are defined in [Section 13.6](#).

**Table 2-7. Serial Flash Memory Interface Signals**

Signal	Ball #	Type	Description
FLSH_SI	P42	t/s	Serial data output that should be connected to the Serial Input (SI) of the SPI serial Flash memory.
FLSH_SO	T42	In Pu	Serial data input that should be connected to the Serial Output (SO) from the SPI serial Flash memory.
FLSH_SCK	P40	t/s	Flash serial clock operates at 25 MHz.
FLSH_CE_N	R41	t/s	Flash chip select output.

2.2.6 General Purpose I/O (GPIO) Pins

This section provides the pin assignment for GPIO signals. The X710-TM4 has a total of 30 GPIO pins that can be configured as Software Definable Pins (SDPs), LED drivers or dedicated hardware functions for connecting to external PHYs or IEEE 1588 auxiliary devices. The GPIO pins can also be associated with any of the physical ports. The following sections show the default configuration for GPIO pins that are reserved for specific use and hence named by default as SDP, LED or GPIO signals. However, the X710-TM4 offers the flexibility to configure any of the GPIO pins (irrespective of the name) to different modes and associated with different ports as described in [Section 3.4.11](#).

The AC/DC specifications for the GPIO signals are defined in [Section 13.6](#).

2.2.6.1 LED Interface Pins

This section provides the pin assignment for LED interface signals. These are GPIO signals that are configured by default as LED interface pins associated with physical ports 0-3. The mode and port association for these pins can be configured (or changed) as described in [Section 3.4.11](#).

The AC/DC specifications for the GPIO/LED signals are defined in [Section 13.6](#).

Table 2-8. LED Interface Signals

Note: Refer to [Table 2-14](#) for specific pins that apply to the X710-AT2.

Signal	Ball #	Type	Description
LED0_0	V4	O	Port 0 LED0. Programmable LED indicates link up and activity (default).
LED0_1	V6	O	Port 0 LED1. Programmable LED indicates 10 Gb/s link (default).
LED1_0	U5	O	Port 2 LED0. Programmable LED indicates link up and activity (default).
LED1_1	U7	O	Port 2 LED1. Programmable LED indicates 10 Gb/s link (default).
LED2_0	T4	O	Port 1 LED0. Programmable LED indicates link up and activity (default).
LED2_1	T6	O	Port 1 LED1. Programmable LED indicates 10 Gb/s link (default).
LED3_0	R3	O	Port 3 LED0. Programmable LED indicates link up and activity (default).
LED3_1	R5	O	Port 3 LED1. Programmable LED indicates 10 Gb/s link (default).



2.2.6.2 SDP Interface Pins

This section provides the pin assignment for SDP interface signals. These are GPIO signals configured by default as SDP pins associated with physical ports 0-3. The mode and port association for these pins can be configured (or changed) as described in [Section 3.4.11](#).

The AC/DC specifications for the GPIO signals are defined in [Section 13.6](#).

Table 2-9. SDP Pins

Note: Refer to [Table 2-14](#) for specific pins that apply to the X710-AT2.

Signal	Ball #	Type	Description
SDP0_0 SDP0_1 SDP0_2 SDP0_3	E5 G5 K6 L5	t/s Pu	SDPs. General purpose 3.3V I/Os. Can be used to connect IEEE1588 auxiliary devices, low speed optical module interfaces, external PHY control or other similar usages. The SDP pins can also be configured for use as external interrupt sources.
SDP1_0 SDP1_1 SDP1_2 SDP1_3	R7 P4 P6 H6	t/s Pu	SDPs. General purpose 3.3V I/Os. Can be used to connect IEEE1588 auxiliary devices, low speed optical module interfaces, external PHY control or other similar usages. The SDP pins can also be configured for use as external interrupt sources.
SDP2_0 SDP2_1 SDP2_2 SDP2_3	F6 J7 E7 G7	t/s Pu	SDPs. General purpose 3.3V I/Os. Can be used to connect IEEE1588 auxiliary devices, low speed optical module interfaces, external PHY control or other similar usages. The SDP pins can also be configured for use as external interrupt sources.
SDP3_0 SDP3_1 SDP3_2 SDP3_3	N5 N7 M6 M8	t/s Pu	SDPs. General purpose 3.3V I/Os. Can be used to connect IEEE1588 auxiliary devices, low speed optical module interfaces, external PHY control or other similar usages. The SDP pins can also be configured for use as external interrupt sources.

2.2.6.3 Global GPIO Interface Pins

This section provides the pin assignment for Global GPIO interface signals. These are GPIO signals that are not assigned for a specific use or port. The mode and port association for these pins can be configured as described in [Section 3.4.11](#).

The AC/DC specifications for the GPIO signals are defined in [Section 13.6](#).

Table 2-10. Global GPIO Pins

Signal	Ball #	Type	Description
GPIO_0 GPIO_1 GPIO_2 GPIO_3	V40 W41 Y42 V42	t/s Pu	General purpose 3.3V I/Os. Can be configured to be associated with any of the ports 0 to 3. Can be used as LED interface or SDP or to connect IEEE1588 auxiliary devices, low speed optical module interfaces, external PHY control or other similar usages. The pins can also be configured for use as external interrupt sources.
GPIO_4 GPIO_5	H8 L7	t/s Pu	General purpose 3.3V I/Os. Can be configured to be associated with any of the ports 0 to 3. Can be used as LED interface or SDP or to connect IEEE1588 auxiliary devices, low speed optical module interfaces, external PHY control or other similar usages. The pins can also be configured for use as external interrupt sources.



2.2.7 Miscellaneous Pins

This section provides the pin assignment for other miscellaneous signals. The AC/DC specifications for the miscellaneous signals are defined in [Section 13.6](#).

Table 2-11. Miscellaneous Pins

Signal	Ball #	Type	Description
TPIN_3	H20		Port 1 PHY MDI pair swap configuration strap.
TPIN_5	G17		Port 0 PHY MDI pair swap configuration strap.
XTAL_PTP_XTAL_N XTAL_PTP_XTAL_P	H42 J41		
LAN_PWR_GOOD	AB42	In Pu	LAN Power Good. A 3.3V input signal. A transition from low-to-high initializes the X710-TM4 into operation. If not used (POR_BYPASS = 0b), an internal Power-on-Reset (POR) circuit triggers the X710-TM4 power-up. If the internal POR circuit is used to trigger device power-up, this signal should be connected to 3.3V. By default, internal POR should be used.
POR_BYPASS	U39	In Pu	Bypass indication as to whether or not to use the internal POR or the LAN_PWR_GOOD pin. When set to 1b, the X710-TM4 disables the internal POR circuit and uses the LAN_PWR_GOOD pin as a POR indication. When set to 0b, the X710-TM4 uses the internal POR circuit. By default, the internal POR should be used unless the power supply sequencing timing requirements, as defined in Section 13.0 , could not be met.
OSC_SEL	J9	t/s Pu	Defines the input clock connected to the REFCLKIN_p_XTAL_IN / REFCLKIN_n_XTAL_OUT pins: 0b = XTAL clock. 1b = OSC clock. This pin is a strapping option latched at LAN_PWR_GOOD.
AUX_PWR	E1	t/s	Auxiliary Power Available. When set, indicates that auxiliary power is available and the X710-TM4 should support the D3 _{COLD} power state if enabled to do so. This pin is latched at the rising edge of LAN_PWR_GOOD.
MAIN_PWR_OK	E3	In	Main Power OK. Indicates that platform main power is up. Must be connected externally.
PCI_DIS_N	U3	t/s Pu	This pin is a strapping pin latched while LAN_PWR_GOOD or PE_RST_N or In-band PCIe reset are asserted. If this pin is not connected or driven high during initialization, then all PCI functions as configured from NVRAM are enabled. If this pin is asserted/driven low during initialization, then all PCI functions that are allowed to be disabled as configured in NVRAM are disabled (see Section 4.5 for details).
DEV_DIS_N	D2	t/s Pu	This pin is a strapping option pin latched while LAN_PWR_GOOD or PE_RST_N or In-band PCIe reset are asserted. This pin can be either used as a device disable or for disabling the LAN ports and associated functions based on NVRAM configuration (See Section 4.5 for details). If this pin is not connected or driven high during initialization, then all the LAN ports and associated functions as configured from NVRAM are enabled for normal operation. If this pin is asserted/driven low during initialization then the LAN ports and associated functions as configured from NVRAM are disabled. Asserting this pin disables the entire device if all the LAN ports are configured to be disabled. When the entire device is disabled, the PCIe link is in L3 state, the PHY is in power down mode, and the output buffers are tri-stated. (see Section 4.5 for details).
RBIAS RSENSE	Y10 Y12	B	BIAS. A 4.75 K Ω \pm 0.1% resistor should be connected between the RBIAS and RSENSE pins. Connect a resistor as close as possible to the chip. A resistor is used for internal impedance compensation and BIAS current generation circuitry.



Signal	Ball #	Type	Description
REFCLK_SEL	D4	In Pd	Select Reference Clock. Active-high. Pulling this pin high enables the XTAL_N/P inputs to select a 156.25 MHz reference input source. Pulling this pin low enables a 50 MHz reference clock input source.
RESET_N	D12	In Pu	Reset. Active-low. Schmitt Trigger input. This signal resets port 0 and port 1. It requires a hardware reset prior to normal operation. Configuration settings obtained through hardware strap option pins are latched on the rising edge.
SYNCE_LOCK_0 SYNCE_LOCK_1 SYNCE_LOCK_2 SYNCE_LOCK_3	F8 E9 E11 F12	O	Synchronous Ethernet Lock. Reserved. Do not use.
SYNCE_CLKOUT_0 SYNCE_CLKOUT_1 SYNCE_CLKOUT_2 SYNCE_CLKOUT_3	G9 F10 G11 E13	O	Synchronous Ethernet Recovered Clock. Reserved. Do not use.
RESCAL_RES_P	J37		BASE-T calibration resistor.
BIAS_RDAC	E39		BASE-T bias resistor.
P0_INT_MLC P1_INT_MLC	M40 F14		BASE-T Port Interrupts.
P0_PTP_SYNC0 P0_PTP_SYNC1	N39 N41		PTP Synchronous Ethernet pins. Reserved. Do not use.
P1_PTP_SYNC0 P1_PTP_SYNC1	H10 H12		PTP Synchronous Ethernet pins. Reserved. Do not use.
XTAL_BYPASS	D10		Clock source configuration pin.

2.2.8 Testability and Debug Pins

This section provides the pin assignment for JTAG testability interface signals. The AC/DC specifications for the JTAG testability signals are defined in [Section 13.6](#).

Table 2-12. Testability and Debug Pins

1.88V PHY I/O:

Signal	Ball #	Type	Description
PHY_TCK	E17	In	BASE-T PHY JTAG Clock Input.
PHY_TDI	E15	In Pu	BASE-T PHY JTAG Data Input.
PHY_TDO	F16	o/d	BASE-T PHY JTAG Data Output.
PHY_TMS	D16	In Pu	BASE-T PHY JTAG TMS Input.
PHY_TRST_N	D18	In Pu	BASE-T PHY JTAG Reset Input. Active low.
DEBUG_W7	W7		
DEBUG_Y16	Y16		
DEBUG_Y20	Y20		



3.3V Core I/O:

Signal	Ball #	Type	Description
JTCK	U37	In	JTAG Clock Input.
JTDI	T36	In Pu	JTAG Data Input.
JTDO	T38	o/d	JTAG Data Output.
JTMS	V38	In Pu	JTAG TMS Input.
JRST_N	V36	In Pu	JTAG Reset Input. Active low reset for the JTAG port.

2.2.9 Reserved and No-connect Pins (X710-TM4)

This section provides the pin assignment for reserved and no-connect pins for the X710-TM4.

Table 2-13. Reserved and No-connect Pins

Signal	Ball #	Type	Description
RSVDD14_NC	D14	In Pu	Reserved.
RSVDR33_NC	R33	O	Reserved.
RSVDP34_NC	P34		Reserved.
RSVDM34_NC	M34		Reserved.
RSVDN33_NC	N33	O	Reserved.
RSVDL33_NC	L33	Pd	Reserved.
RSVDL31_VSS	L31		Reserved.
RSVDF38_VSS	F38		Reserved.
RSVDK30_NC RSVDK32_NC	K30 K32		Reserved.
RSVDK34_NC RSVDK36_NC	K34 K36		Reserved.
RSVDK26_NC RSVDK28_NC	K26 K28		Reserved. Reserved.
RSVDD32_NC	D32		Reserved.



Signal	Ball #	Type	Description
RSVDL29_NC RSVDM32_NC RSVDM30_NC RSVDM31_NC RSVDN29_NC RSVDP32_NC RSVDP30_NC RSVDR31_NC RSVDR29_NC RSVDT32_NC RSVDT30_NC RSVDU35_NC RSVDU33_NC RSVDU31_NC RSVDV32_NC RSVDV30_NC	L29 M32 M30 N31 N29 P32 P30 R31 R29 T32 T30 U35 U33 U31 V32 V30		Reserved.
RSVD_J11_VSS	J11		Reserved.
RSVDV8_NC RSVDV10_NC	V8 V10		Reserved. Reserved.
RSVDH40_VSS RSVDK40_1P88V RSVDK42_VSS RSVDP38_VSS RSVDN37_VSS RSVDL37_VSS	H40 K40 K42 P38 N37 L37	In Pd	Reserved.
RSVDA41_NC RSVDB42_NC	A41 B42		Reserved. Reserved.
RSVDC21_NC RSVDD20_NC	C21 D20		Reserved. Reserved.
RSVDG23_VSS RSVDH24_VSS	G23 H24		Reserved. Reserved.
RSVDJ13_VSS	J13		Reserved.
RSVDJ39_NC	J39		Reserved.
RSVDJ17_VSS	J17		Reserved.
RSVDD8_1P88V	D8		Reserved.
RSVDD6_VSS	D6		Reserved.
RSVDL35_NC RSVDM36_NC RSVDN35_NC RSVDP36_NC RSVDR37_NC	L35 M36 N35 P36 R37	O	Reserved.
RSVDH16_NC RSVDG15_NC RSVDJ15_NC RSVDH14_NC RSVDG13_NC	H16 G15 J15 H14 G13	O	Reserved.



Signal	Ball #	Type	Description	
RSVDN23_NC RSVDR27_NC RSVDW23_NC RSVDM22_NC RSVDP28_NC RSVDR25_NC RSVDV26_NC	N23 R27 W23 M22 P28 R25 V26		Reserved.	
RSVDK22_NC RSVDM28_NC RSVDN27_NC RSVDP26_NC RSVDR23_NC RSVDT26_NC RSVDT22_NC RSVDU27_NC RSVDU23_NC	K22 M28 N27 P26 R23 T26 T22 U27 U23			
RSVDV24_NC RSVDW27_NC RSVDJ25_NC RSVDL27_NC RSVDM26_NC RSVDN25_NC RSVDP24_NC RSVDT28_NC RSVDT24_NC	V24 W27 J25 L27 M26 N25 P24 T28 T24			
RSVDU29_NC RSVDU25_NC RSVDV28_NC RSVDV22_NC RSVDW25_NC RSVDJ23_NC RSVDL23_NC RSVDM24_NC	U29 U25 V28 V22 W25 J23 L23 M24			
RSVDE25_VSS RSVDD22_VSS RSVDH22_VSS RSVDE21_VSS RSVDF22_VSS RSVDF18_VSS RSVDG19_VSS RSVDJ21_VSS RSVDE23_VSS RSVDE19_1P88V RSVDF20_VSS RSVDG21_VSS RSVDG17_VSS RSVH18_VSS RSVDJ19_VSS	E25 D22 H22 E21 F22 F18 G19 J21 E23 E19 F20 G21 G17 H18 J19			
RSVDY18_VSS	Y18			Reserved.



Signal	Ball #	Type	Description
RSVDP22_VSS RSVDY22_VSS	P22 Y22		Reserved.
RSVDN9_VSS RSVDW5_VSS RSVDW1_VSS RSVDU9_NC RSVDR9_NC RSVDL25_NC RSVK24_NC	N9 W5 W1 U9 R9 L25 K24		
DEBUG_Y20 DEBUG_W7 DEBUG_Y16	Y20 W7 Y16		
RSVDP8_VSS RSVDY24_VSS	P8 Y24		
RSVDW3_VSS	W3		

2.2.10 No-connect Pins (X710-AT2)

This section provides the no-connect pin assignments for the X710-AT2.

Table 2-14. No-connect Pins X710-AT2

Signal	Ball #	Type	Description
SDP0_3	L5	t/s Pu	
SDP1_0 SDP1_1 SDP1_2 SDP1_3	R7 P4 P6 H6	t/s Pu	
SDP2_0 SDP2_1 SDP2_2 SDP2_3	F6 J7 E7 G7	t/s Pu	
SDP3_0 SDP3_1 SDP3_2 SDP3_3	N5 N7 M6 M8	t/s Pu	
MDIO1_SDA1 MDIO2_SDA2 MDIO2_SDA2	M2 L1 K2	t/s o/d	



Signal	Ball #	Type	Description
MDC1_SCL1 MDC2_SCL2 MDC2_SCL3	M4 L3 K4	O o/d	
LED3_0 LED3_1 LED1_0 LED1_1 LED3_0 LED3_1	U5 U7 R3 R5	O	
GPIO_1 GPIO_3 GPIO_4	W41 V42 E5 H8	t/s Pu	

2.2.11 Integrated Voltage Regulator (SVR) Pins

This section provides the pin assignment for SVR pins. The electrical specifications of the SVR pins are defined in [Section 13.6](#).

Table 2-15. Integrated SVR Pins

Signal	Ball #	Type	Description
SVR_IND	V2		Internal node of the integrated SVR. Connect to the analog power supply (VCCA) rail through an external inductor.
VCC3P3_SVR	T2	3.3V	Power supply input to integrated SVR (analog voltage).
VSS_SVR	U1	0V	Integrated SVR GND (analog voltage).
VSS_S	Y2	0V	Integrated SVR GND (analog voltage). Internal shield around the power switch.
RSVDT8_VCCD	T8		Reserved.

2.2.12 Power Supply Pins

This section provides the pin assignment for power supply pins. The electrical specifications for the power supply pins are defined in [Section 13.6](#).

Table 2-16. Power Supply Pins

Signal	Ball #	Type	Description
XFI_PVDD	J33	1.0V	Digital power supply.
XFI_VDD	J27, J29, J31, J35	1.0V	Digital power supply.
XGB_AVDDH	D34	1.88V	Digital power supply.
BIAS_AVDDH	D30	1.88V	Digital power supply.



Signal	Ball #	Type	Description
VDDIO1	G25, G37	1.88V	
VDDIO2	L39	1.88V	Connect to a 1.2V supply when MDIO_MODE is set low. Connect to a 2.5V supply when MDIO_MODE is high.
VDDP2	M38	1.88V	
DVDD	G27, G29, G31, G33, G35	1.0V	Digital power supply.
VCC3P3	U21, R11, T34	3.3V	Digital power supply.
AVSS	A21, B22, D24, A31, B32, C31, D26, D28, D36, E31, E33, C41, D38, D40, F40, F42, G41		
VCC3P3_A	W11	3.3V	Analog power supply.
VCC3P3_TX	L9	3.3V	Power supply for high speed SerDes transmit interfaces.
VCCD	L19, L21, N17, N19, N21, R13, R15, R17, R19, R21, U11, U13, U15, U17, U19	0.92V	Digital power supply.
VCCD_A	W9	0.92V	Analog power supply.
P0_AVDDL P1_AVDDL	C33, C35, C37, C39 C23, C25, C27, C29	1.0V	
P0_AVDDH P1_AVDDH	E35, E37 E27, E29	1.88V	
VCCA	L11, L13, L15, L17, N11, N13, N15, W13, W15, W17, W19, W21	0.935V	Analog power supply.
VSSA	A3, A7, A11, B2, B4, B8, B12, B16, C5, C7, C9, C11, C13, C15, C17, C19, H26, H28, H30, H32, H34, H36, H38, K8, K10, K12, M10, M12, V12, Y4, Y6, Y8, AA3, AA9, AA11, AB4, AB6, AB8, AC1, AC3, AC7, AC9, AC11, AD2, AD6, AD12, A15, A19, B20, K14, K16, M14, M16, V14, V16, V18, V20, Y14, AA13, AA15, AA17, AA19, AA21, AA23, AB14, AB20, AC13, AC15, AC17, AC19, AC21, AC23, AD18, AD24, W29, W31, W33, W35, Y26, Y28, Y32, Y36, AA25, AA29, AA33, AB26, AB30, AB32, AB34, AB36, AC25, AC27, AC29, AC33, AD30, AD34, K38, W37, W39, Y40, AA37, AA41, AB38, AB40, AC37, AC41, AD38, AD42	0V	Analog power supply ground.
VSS	F24, F26, F28, F30, F32, F34, F36, K18, K20, M18, M20, P10, P12, P14, P16, P18, P20, R35, T10, T12, T14, T16, T18, T20	0V	Digital power supply ground.
PLL_VDD	G39	1.0V	



2.2.13 Pull-up and Pull-down Resistors

Internal pull-up values:

- Min: 54 K Ω
- Typ: 74 K Ω
- Max: 110 K Ω

Table 2-17. Pull-up and Pull-down Resistors

Pin Name	Internal	External	Comments
PE_WAKE_N		Pup	The pull-up resistor exists in the main platform as part of the system wake signal.
PE_RST_N	Pup		
MDIO0_SDAx		Pup 3.3 K Ω	x=0,...,3.
MDC0_SCLx		Pup 3.3 K Ω	x=0,...,3.
NCSI_CLK_IN	Pup	Pdn 100 K Ω	
NCSI_CRS_DV	Pup	Pdn 100 K Ω	
NCSI_RXD_0 NCSI_RXD_1		Pup 100 K Ω Pup 100 K Ω	
NCSI_TX_EN	Pup	Pdn 100 K Ω	
NCSI_TXD_0 NCSI_TXD_1	Pup Pup	Pup 100 K Ω Pup 100 K Ω	
NCSI_ARB_IN	Pup	Pdn 10 K Ω	
NCSI_ARB_OUT	Pup		
SMBCLK		Pup 10 K Ω	
SMBD		Pup 10 K Ω	
SMBALRT_N		Pup 10 K Ω	
FLSH_SO	Pup		
FLSH_CE_N		Pup 10 K Ω	
SDPx_y	Pup		
LAN_PWR_GOOD	Pup	Pup 10 K Ω	
POR_BYPASS		Pdn 100 Ω	
OSC_SEL	Pup	See comment	Connect to VSS / VCC if using a crystal or oscillator, respectively.
AUX_PWR	Pup	See comment	Connect the AUX_PWR signal to a 10 K Ω pull-up resistor if AUX power is available. Connect a pull-down resistor (0 Ω) if AUX power is not available.
MAIN_PWR_OK	Pup	See comment	Connect MAIN_PWR_OK signal to main power through a pull-up resistor. A pull-up value should be considered as 10 K Ω .
PCI_DIS_N	Pup	Pup 10 K Ω	
DEV_DIS_N	Pup	Pup 10 K Ω	
JTCK		Pdn 470 Ω	
JTDO		Pup 8.2 K Ω	



Pin Name	Internal	External	Comments
JRST_N		Pdn 10 K Ω	
RSVDx_NC			x=0,...,31.
RSVDN9_VSS RSVDW5_VSS RSVDW1_VSS	Pup Pup Pup		
RSVDY20_VSS RSVDW7_VSS RSVDY16_VSS DEBUG_Y2 0 DEBUG_W7 DEBUG_Y16	Pup Pup Pup	Pdn 100 Ω Pdn 100 Ω Pdn 100 Ω	
RSVDY24_VSS	Pup		



2.3 Package Layout

Figure 2-1 shows a top view ball map of the X710-TM4, in a 22 mm x 22 mm 503-pin FCBGA package.

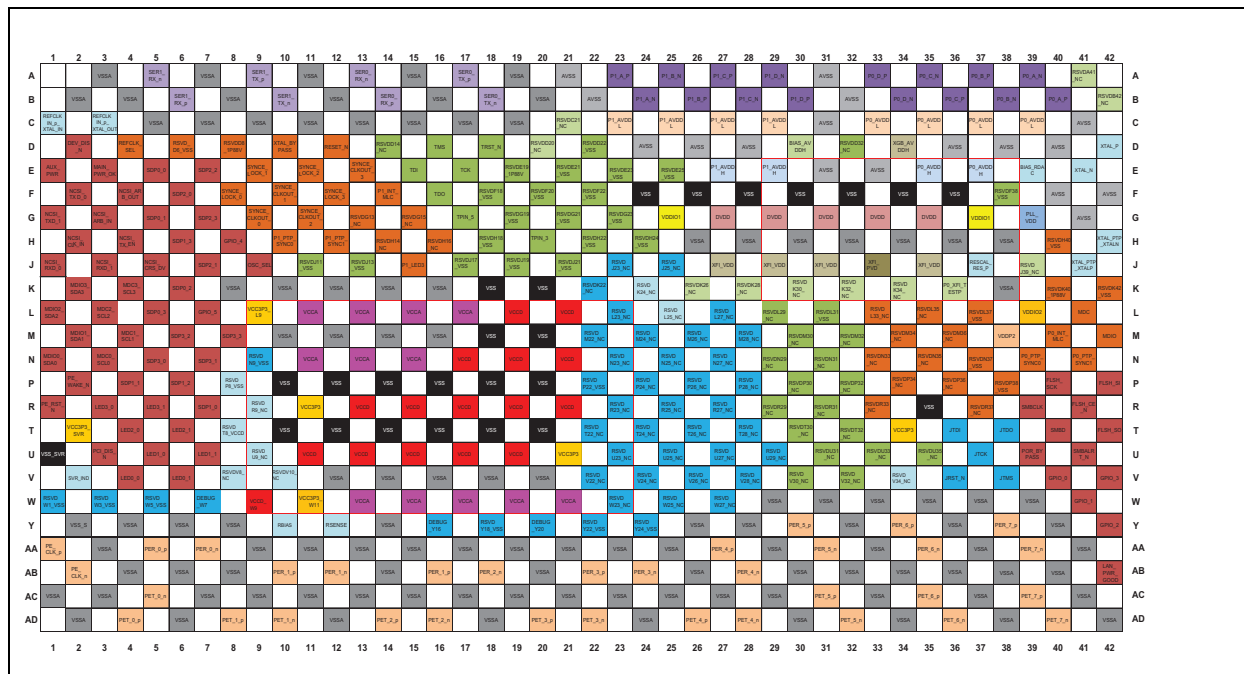


Figure 2-1. Package Layout

Note: Refer to Table 2-14 in Section 2.2.10 for pins that are not connected for the X710-AT2.



NOTE: *This page intentionally left blank.*



3.0 Interconnects

3.1 PCI-Express* (PCIe*)

3.1.1 Requirements

The X710-TM4 supports Rev. 3.0 of the PCIe base specification.

In addition to the capabilities required by the PCIe specifications, the X710-TM4 also supports the following optional functionality as described in this section:

- All PCI functions are native PCIe functions
- Physical Layer
 - Support for 2.5 GT/s, 5 GT/s, and 8 GT/s
 - Interface width of 1, 4, or 8 PCIe lanes
 - Full swing and half swing signaling
 - Lane reversal
- Transaction layer mechanisms
 - 64-bit and 32-bit memory address spaces
 - Removal of I/O BAR (optional)
 - Relaxed ordering
 - Flow control update timeout mechanism
 - ID-based ordering (IDO)
 - Packet sizes: Maximum payload size: 512 bytes, Maximum read request size: 4 KB
 - Extended tags
 - Function-Level Reset (FLR)
- Reliability
 - Advanced Error Reporting (AER)
 - End-to-End CRC (ECRC) generation and checking
 - Recovery from data poisoning
 - Completion timeout
- Power management:
 - Active state power management (L0s and L1 states)
 - Wake capability
- DFT and DFM support for high-volume manufacturing
- The X710-TM4 supports the following extended capabilities:
 - AER
 - Device Serial Number (DSN)
 - Alternative RID Interpretation (ARI)
 - Single Root I/O Virtualization (SR-IOV)
 - Access Control Services (ACS)
 - TLP Processing Hints (TPH)



3.1.2 Transaction Layer

3.1.2.1 Transactions Accepted by the X710-TM4

Table 3-1 lists the transactions accepted by the device and their attributes. See Section 3.1.2.8 for the number of credits provided per FC type.

Table 3-1. Transaction Types Accepted by the Transaction Layer

Transaction Type	FC Type	Tx Layer Reaction	Hardware Should Keep Data From Original Packet
Configuration Read Request	NPH	CPLH + CPLD	Requester ID, TAG, attribute.
Configuration Write Request	NPH + NPD	CPLH	Requester ID, TAG, attribute.
Memory Read Request	NPH	CPLH + CPLD	Requester ID, TAG, attribute.
Memory Write Request	PH + PD	-	-
IO Read Request	NPH	CPLH + CPLD	Requester ID, TAG, attribute.
IO Write Request	NPH + NPD	CPLH	Requester ID, TAG, attribute.
Read Completions	CPLH + CPLD	-	-
Message	PH + PD ¹	-	-

1. MCTP messages contain payload.

Flow Control Types Legend:

- CPLD — Completion Data Payload
- CPLH — Completion Headers
- NPD — Non-Posted Request Data Payload
- NPH — Non-Posted Request Headers
- PD — Posted Request Data Payload
- PH — Posted Request Headers

3.1.2.2 Size of Target Accesses

3.1.2.2.1 Memory Accesses

Rules for accesses to the CSR space (both memory BAR and MSI-X BAR):

- Write accesses:
 - CSR writes are 32 bit or 64 bit only
 - Zero-length writes have no internal impact (nothing written, no effect such as clear-by-write). The transaction is treated as a successful operation (no error event)
 - Other accesses (partial writes, larger writes) are handled as completer abort - data is dropped and an error is generated per PCIe rules



- Read accesses:
 - CSR reads are 32 bit or 64 bit only
 - Some 64-bit reads are handled atomically such as not interleaved with any other read requests. This applies mainly to reading counters, where all 64 bits need to be read simultaneously. Such registers are explicitly marked in their description.
 - Partial reads with at least one byte disabled are handled internally as a full read. That is, any side effect of the full read (such as clear by read) is also applicable to partial reads. The completion on PCIe adheres to the specification rules regarding the number of bytes reported in the completion.
 - Zero-length reads generate a completion, but the register is not accessed and undefined data is returned
 - Larger CSR read requests are handled as completer abort - the completion includes a CA status and an error is generated per PCIe rules

Rules for accessing the Flash space in the memory BAR or the expansion ROM BAR:

- Write accesses:
 - Writes to Flash are silently ignored
- Read accesses:
 - Reads to Flash are 32-bit wide
 - Partial reads with at least one byte disabled are handled internally as a full read. The completion on PCIe adheres to the specification rules regarding the number of bytes reported in the completion
 - Zero-length reads generate a completion, but the Flash is not accessed and undefined data is returned
 - Larger CSR read requests are handled as completer abort - the completion includes a CA status and an error is generated per PCIe rules

3.1.2.3 I/O Accesses

Rules for accesses to the I/O BAR:

- Write accesses:
 - Write accesses are 32-bit wide
 - Zero-length writes have no internal impact (nothing written, no effect such as clear-by-write). The transaction is treated as a successful operation (no error event)
 - Other accesses (partial writes, larger writes) are handled as completer abort - data is dropped and an error is generated per PCIe rules
- Read accesses:
 - Reads to the I/O BAR are 32-bit wide
 - Partial reads with at least one byte disabled are handled internally as a full read. That is, any side effect of the full read (such as clear by read) is also applicable to partial reads. The completion on PCIe adheres to the specification rules regarding the number of bytes reported in the completion
 - Larger CSR read requests are handled as completer abort - the completion includes a CA status and an error is generated per PCIe rules
 - See [Section 10.1.1.5](#) for more details.



3.1.2.3.1 Messages

MCTP messages might contain a payload of up to 64 bytes.

3.1.2.4 Transactions Initiated by the X710-TM4

Table 3-2 lists the transactions initiated by the device and their attributes.

Table 3-2. Transaction Types Initiated by the Transaction Layer

Transaction type	Payload Size	FC Type
Configuration Read Request Completion	Dword	CPLH + CPLD
Configuration Write Request Completion	-	CPLH
IO Read Request Completion	Dword	CPLH + CPLD
IO Write Request Completion	-	CPLH
Read Request Completion	Dword/Qword	CPLH + CPLD
Memory Read Request	-	NPH
Memory Write Request	≤ MAX_PAYLOAD_SIZE	PH + PD
Message	≤ 64 bytes ¹	PH + PD

1. MCTP messages contain payload.

Configuration values:

- Max Payload Size - The value of the Max_Payload_Size Supported field in the Device Capabilities register is loaded from NVM.
 - Hardware default is 512 bytes.
 - System software then programs the actual value into the Max_Payload_Size field of the Device Control register.
 - Non-ARI mode: If not all functions are programmed with the same value, the max payload size used for all functions is the minimum value programmed among all functions.
 - ARI mode: Max_Payload_Size is determined solely by the setting in Function 0
- Max_Read_Request_Size - The X710-TM4 supports read requests of up to 4 KB.
 - The actual maximum size of a read request is defined as the minimum {4 KB, Max_Read_Request_Size field in the Device Control Register}.
- Extended tags are supported for Memory Read Requests.

3.1.2.4.1 Data Alignment

Requests must never specify an address/length combination that causes a memory space access to cross a 4 KB boundary. The X710-TM4 therefore breaks requests into 4 KB-aligned requests (if needed). This does not pose any requirement on software. However, if software allocates a buffer across a 4 KB boundary, hardware issues multiple requests for the buffer. Software should consider aligning buffers to a 4 KB boundary in cases where it improves performance.



The general rules for packet alignment are as follows. Note that these apply to all the X710-TM4 requests:

- The length of a single request does not exceed the PCIe limit of MAX_PAYLOAD_SIZE for write and MAX_READ_REQUEST_SIZE for read.
- A single request does not span across different memory pages as noted by the 4 KB boundary alignment previously mentioned.

If a request can be sent as a single PCIe packet and still meets the general rules for packet alignment, then it is not broken at the cache line boundary but rather sent as a single packet. However, if any of the general rules require that the request is broken into two or more packets, then the request is broken at the cache line boundary.

For requests with data payload, if the payload size is larger than (MAX_PAYLOAD_SIZE - CACHELINE_SIZE), then the request is broken into multiple TLPs starting at the first cache-line boundary following the (MAX_PAYLOAD_SIZE - CACHELINE_SIZE) bytes. For example, if MAX_PAYLOAD_SIZE = 256B and CACHELINE_SIZE = 64 bytes, a 1 KB request starting at address 0x...10 is broken into TLPs such that the first TLP contains 240bytes of payload (since 240bytes + 0x10 = 256bytes is on the cache-line boundary)

The system cache line size is controlled by the GLPCI_CNF2.CACHELINE_SIZE bit, loaded from the NVM *Cache Line Size* field. Note that the Cache Line Size register in the PCI configuration space is not related to the GLPCI_CNF2.CACHELINE_SIZE and is solely for software use.

3.1.2.5 Messages

Table 3-3 lists the response to messages sent to the device. Unlisted messages are not supported by the device and are treated as an unsupported request.

Table 3-3. Messages in the X710-TM4 (as a Receiver)

Message Code [7:0]	Routing r2r1r0	Message	X710-TM4 Response
0x00	011b	Unlock	Silently drop.
0x14	100b	PM_Active_State_NAK	Accepted.
0x19	011b	PME_Turn_Off	Accepted.
0x40 0x41 0x43 0x44 0x45 0x47 0x48	100b	Ignored messages (used to be hot-plug messages)	Silently drop.
0x50	100b	Slot power limit support (has one Dword data)	Silently drop.



Table 3-3. Messages in the X710-TM4 (as a Receiver)

Message Code [7:0]	Routing r2r1r0	Message	X710-TM4 Response
0x00	011b	Unlock	Silently drop.
0x7E	000b 010b 011b 100b	Vendor_defined type 0	Drop and handle as an unsupported request.
0x7F	100b 000b	Vendor_defined type 1	Silently drop.
0x7F	010b 011b	Vendor_defined type 1 See Section 3.1.2.5.1	Send to MCTP reassembly if Vendor ID = 0x1AB4 (DMTF) and VDM code - 0000b (MCTP). Otherwise, silently drop.

Table 3-4 lists the messages sent by the device.

Table 3-4. Messages in the X710-TM4 (as a Transmitter)

Message code [7:0]	Routing r2r1r0	Message
0x20	100b	Assert INT A.
0x21	100b	Assert INT B.
0x22	100b	Assert INT C.
0x23	100b	Assert INT D.
0x24	100b	De-assert INT A.
0x25	100b	De-assert INT B.
0x26	100b	De-assert INT C.
0x27	100b	De-assert INT D.
0x30	000b	ERR_COR.
0x31	000b	ERR_NONFATAL.
0x33	000b	ERR_FATAL.
0x18	000b	PM_PME.
0x1B	101b	PME_TO_Ack.
0x7F	000b 010b 011b	Vendor Defined Messages (VDM); see Section 3.1.2.5.1 .

3.1.2.5.1 VDM

The following vendor defined message is supported: DMTF MCTP.

3.1.2.5.1.1 MCTP VDMs

MCTP VDMs are supported as both master and target. The following header fields are involved (see [Section 9.7.3.1](#) for more details):

- Fmt - Set to 11b to indicate a 4-Dword header with data



- Type:
 - [4:3] - Set to 10b to indicate a message
 - [2:0] - Routing r2r1r0 = 000b, 010b or 011b
- Traffic Class - Set to 000b
- TLP Digest - Set to 0b (no ECRC)
- Error Present - Set to 0b
- Attributes[1:0] - Set to 01b (no snoop)
- Tag field - Indicates this is an MCTP packet and the size of padding to Dword alignment added
- Message code = 0x7F (Type 1 VDM)
- Destination ID - captures the target B/D/F for route by ID. Otherwise, reserved
- Vendor ID = 0x1AB4 (DMTF)

3.1.2.6 Transaction Attributes

3.1.2.6.1 Traffic Class (TC) and Virtual Channels (VC)

The X710-TM4 only supports TC = 0b and VC = 0b (default).

3.1.2.6.2 TLP Processing Hints (TPH)

The X710-TM4 supports the TPH capability defined in the PCI Express specification. It does not support extended TPH requests.

Existence of a TLP Process Hint (TPH) is indicated on the PCIe link by setting the TH bit in the TLP header. Using the PCIe TLP Steering Tag (ST) and Processing Hints (PH) fields, the X710-TM4 can provide hints to the root complex about the destination (socket ID) and about data access patterns (locality in cache) when executing DMA memory writes or read operations.

The X710-TM4 exposes a PCIe TPH capability structure (see [Section 11.4.5](#)) with no steering table.

Required steps to enable TPH usage:

1. For a given function, the *TPH Requester Enable* field in the PCIe configuration TPH Requester Control register should be set to either 01b or 11b and the *ST Mode Select* field should be set to one of the two supported values: 000b (No Table Mode) or 010b (Device Specific Mode). If this is not the case, the PF driver should not enable the TPH in the transmit and receive queue contexts.
2. Appropriate TPH enable bits in the receive or transmit queue context should be set.
3. Processing hints should be programmed in the *GLTPH_CTRL.Desc_PH* and *GLTPH_CTRL.Data_PH* Processing Hints (PH) fields.
4. Steering information should be programmed in the CPUID fields in the receive or transmit queue context.

The Processing Hints (PH) and Steering Tags (ST) are set according to the characteristics of the traffic as listed in [Table 8-4](#).

Note: In order to enable TPH usage, all the memory reads are done without setting any of the byte enable bits.



3.1.2.6.2.1 Steering Tag and Processing Hint Programming

Each type of DMA traffic uses a different policy to define how the steering tag (socket ID) and processing hints are generated:

- The policy for LAN traffic is described in [Section 8.2.4](#).
- Accesses to the host memory cache do not use TPH hints.
- Accesses to the Admin command queues do not use TPH hints.

3.1.2.7 Device Ordering Rules

The X710-TM4 meets the PCIe ordering rules as follows:

Deadlock avoidance – The X710-TM4 meets the PCIe ordering rules that prevent deadlocks:

1. Posted writes overtake stalled read requests. This applies to both target and master directions. For example, if master read requests are stalled due to lack of credits, master posted writes are allowed to proceed. On the target side, it is acceptable to timeout on stalled read requests in order to allow later posted writes to proceed.
2. Target posted writes overtake stalled target configuration writes.
3. Completions overtake stalled read requests. This applies to both target and master directions. For example, if master read requests are stalled due to lack of credits, completions generated by the X710-TM4 are allowed to proceed.

Consistency of data:

1. Descriptor/Data Ordering — The X710-TM4 insures that a Rx descriptor is written back on PCIe only after the data that the descriptor relates to is written to the PCIe link.
2. Target NP read requests might pass target posted writes addressing different PCI functions.
3. Completions for target reads (memory, I/O, configuration) do not pass previous posted requests. Here are some specific usages of this rule:
 - Flush following a reset (such as FLR, BME, D3 entry, VFE clear) - When the system issues a reset event, it needs to identify when the device stops sending new posted requests from the function(s) under reset. The completion to the config write of these reset events are sent after all requests/completions related to that function(s) are flushed out. The device is expected not to issue any new posted transactions from the function(s) under reset.
 - MSI and MSI-X Ordering Rules – System software can change the MSI or MSI-X tables during run-time. Software expects that interrupt messages issued after the table has been updated are using the updated contents of the tables.
 - Since software doesn't know when the tables are actually updated in the X710-TM4, a common scheme is to issue a read request to the MSI or MSI-X table after an update to the table (a PCI configuration read for MSI and a memory read for MSI-X). Software expects that any message issued following the completion of the read request, is using the updated contents of the tables.
 - Once an MSI or MSI-X message is issued using the updated contents of the interrupt tables, any consecutive MSI or MSI-X message does not use the contents of the tables prior to the change.



Independence between target and master accesses:

1. The acceptance of a target posted request does not depend upon the transmission of any TLP.
2. The acceptance of a target non-posted request does not depend upon the transmission of a non-posted request.
3. Accepting a completion does not depend upon the transmission of any TLP.

3.1.2.7.1 Processing of Target Accesses

The X710-TM4 meets the specification requirements regarding target accesses as described in the previous section.

In addition, the following behaviors apply:

- Target accesses from different functions might be processed in a different order than the order they arrive
- Completions that belong to requests from different PCI functions might be issued in a different order than the order of the respective requests.

3.1.2.7.2 Relaxed Ordering

The X710-TM4 takes advantage of the relaxed ordering rules in PCIe. By setting the relaxed ordering bit in the packet header, the X710-TM4 enables the system to optimize performance in the following cases:

1. Relaxed ordering for LAN descriptor and data reads — When the X710-TM4 issues a read transaction, its split completion has no ordering relationship with the writes from the CPUs (same direction). It should be allowed to bypass the writes from the CPUs.
 - The *GLLAN_RCTL.RXDESCRDROEN* bit (loaded from NVM) enables relaxed ordering for Rx descriptor reads.
 - The *GLLAN_TCTL.TXDESCRDROEN* bit (loaded from NVM) enables relaxed ordering for Tx descriptor reads.
 - The *GLLAN_TCTL.TXDATARDROEN* bit (loaded from NVM) enables relaxed ordering for Tx data reads.
2. Relaxed ordering for LAN Rx data writes — When the X710-TM4 issues Rx data writes, it also enables them to bypass each other in the path to system memory because software does not process this data until their associated descriptor writes are done.
 - The *GLLAN_RCTL.RXDATAWRROEN* bit (loaded from NVM) enables relaxed ordering for Rx data writes.
3. The X710-TM4 does not relax ordering for the following requests:
 - LAN descriptor writes
 - LAN Tx head write back
 - Interrupt messages
 - MCTP messages
 - HMC requests
 - EMP requests
 - Any other requests not previously mentioned

Relaxed ordering is globally enabled in the X710-TM4 by clearing the *GLPCI_CNF2.RO_DIS* bit, originally loaded from the NVM. It is further controlled through the *Enable Relaxed Ordering* bit in the PCIe Device Control register.



3.1.2.7.3 ID-based Ordering (IDO)

ID-based ordering was introduced in the PCIe rev. 2.1 specification. When enabled, the X710-TM4 sets IDO in all applicable TLPs defined in the PCIe specification. IDO is not set for MCTP packets.

IDO is enabled when all of the following conditions are met:

- IDO is not disabled from the NVM. Device default is enabled. The value loaded from the NVM is reflected in the GLPCI_CAPSUP register.
- The PCIe *IDO Request Enable* bit (for requests) or the *IDO Completion Enable* bit (for completions) in the Device Control 2 register is set.

3.1.2.8 Flow Control

3.1.2.8.1 Flow Control Rules

The X710-TM4 only implements the default Virtual Channel (VC0). A single set of credits is maintained for VC0.

Table 3-5. Flow Control Credits Allocation

Credit Type	Operations	Number of Credits (per device)
Posted Request Header (PH)	Target write (one unit) Message (one unit)	96 header credit units.
Posted Request Data (PD)	Target write Message	288 data credits units.
Non-Posted Request Header (NPH)	Target read (one unit) Configuration read (one unit) Configuration write (one unit)	Four units (to enable concurrent target accesses).
Non-Posted Request Data (NPD)	Configuration write (one unit)	Four units.
Completion Header (CPLH)	Read completion (N/A)	Infinite (accepted immediately).
Completion Data (CPLD)	Read completion (N/A)	Infinite (accepted immediately).

Rules for FC updates:

- UpdateFC packets are sent immediately when a resource becomes available.
- The X710-TM4 follows the PCIe recommendations for frequency of UpdateFC FCPs.
- Specific rules apply in L0 or L0s link state. See the PCIe specification.

3.1.2.8.2 Flow Control Timeout Mechanism

The X710-TM4 implements the optional flow control update timeout mechanism. See the PCIe specification.



3.1.2.9 End-to-End CRC (ECRC)

The X710-TM4 supports ECRC as defined in the PCIe specification. The following functionality is provided:

- Inserting ECRC in transmitted TLPs:
 - The X710-TM4 indicates support for inserting ECRC in the *ECRC Generation Capable* bit of the PCIe Configuration registers. This bit is loaded from the global *ECRC Generation Capable* NVM bit.
 - Inserting ECRC is enabled per function by the *ECRC Generation Enable* bit of the PCIe Configuration registers. VFs follow the behavior of their PF.
 - ECRC is not added to MCTP messages (per the MCTP specification).
- ECRC is checked on all incoming TLPs. A packet received with an ECRC error is dropped. Note that for completions, a completion timeout occurs later (if enabled).
 - The X710-TM4 indicates support for ECRC checking in the *ECRC Check Capable* bit of the PCIe Configuration registers. This bit is loaded from the global *ECRC Check Capable* NVM bit.
 - Checking of ECRC is enabled by the *ECRC Check Enable* bit of the PCIe Configuration registers. ECRC checking is done if enabled by at least one physical function (enablement is not done via VFs).
- ECRC errors are reported on all Physical Functions (PFs) enabled for ECRC checking.
- System software can configure ECRC independently per each physical function.

3.1.3 Link Layer

3.1.3.1 ACK/NAK Scheme

NAKs are sent as soon as identified.

ACKs are sent per section 3.5.3.1 (Table 3-7, Table 3-8, and Table 3-9) in the PCIe Base Specification.

3.1.3.2 Supported DLLPs

The following DLLPs are supported by the X710-TM4 as a receiver:

- ACK
- NAK
- PM_Request_Ack
- InitFC1-P
- InitFC1-NP
- InitFC1-Cpl
- InitFC2-P
- InitFC2-NP



- InitFC2-Cpl
- UpdateFC-P
- UpdateFC-NP
- UpdateFC-Cpl

The following DLLPs are supported by the X710-TM4 as a transmitter:

- ACK
- NAK
- PM_Enter_L1
- PM_Enter_L23
- InitFC1-P
- InitFC1-NP
- InitFC1-Cpl
- InitFC2-P
- InitFC2-NP
- InitFC2-Cpl
- UpdateFC-P
- UpdateFC-NP

Note: UpdateFC-Cpl is not sent because of the infinite FC-Cpl allocation.

3.1.3.3 Transmit EDB Nullifying (End Bad)

A TLP might be signaled as EDB or poisoned if during its transmission from the device, an internal memory error is detected that might corrupt the TLP payload.

3.1.3.4 Retry Buffer

The retry buffer size is 8 KB.

3.1.4 Physical Layer

3.1.4.1 Link Speed

The X710-TM4 supports Gen 1 (2.5GT/s), Gen 2 (5GT/s), Gen 3 (8GT/s).

The following configuration controls link speed:

- *PCIe Supported Link Speeds* bit — Indicates the link speeds supported by the X710-TM4.
- *PCIe Current Link Speed* bit — Indicates the negotiated link speed.
- *PCIe Target Link Speed* bit — used to set the target compliance mode speed when software is using the *Enter Compliance* bit to force a link into compliance mode. The default value is the highest link speed supported defined by the previous *Supported Link Speeds*.



The X710-TM4 does not initiate a hardware autonomous speed change.

The X710-TM4 supports entering compliance mode at the speed indicated in the *Target Link Speed* field in the PCIe Link Control 2 register. Compliance mode functionality is controlled via the PCIe Link Control 2 register.

3.1.4.2 Link Width

The X710-TM4 supports a maximum link width of x8, x4, or x1.

The maximum link width supported is loaded from the NVM into the *Maximum Link Width* field of the PCIe Link Capabilities register. Hardware default is the link.

During link configuration, the platform and the X710-TM4 negotiate on a common link width. The link width must be one of the supported PCIe link widths (x1, x4, x8), such that:

- If maximum link width = x8, then the X710-TM4 negotiates to either x8, x4, or x1
- If maximum link width = x4, then the X710-TM4 negotiates to either x4 or x1
- If maximum link width = x1, then the X710-TM4 only negotiates to x1

The X710-TM4 does not initiate a hardware autonomous link width change.

3.1.4.3 Lane Configurations

The X710-TM4 supports lane reversal and degraded modes.

The following general rules determine how the device reacts in different cases of lanes configuration:

- If lane 0 is found valid, The X710-TM4 does not initiate lane reversal. The Link Partner (LP) might initiate lane reversal (in order to end up with an optimal lane width) and The X710-TM4 consents with the lane reversal.
- If lane 0 is found invalid, The X710-TM4 initiates lane reversal. Lane reversal succeeds if the LP supports link reversal.
- If the lanes at both ends of the port (such as lanes 0 and 7 for x8, lanes 0 and 3 for x4, lane 0 for x1) are invalid, a link is not established.

Note: Some of the configurations or transitions assume lane reversal done by the LP. If the LP does not support a specific transition, then the respective configuration is not provided on that system.

Figure 3-1, Figure 3-2, and Figure 3-3 depict the initial link width configuration and link degradation options. In Figure 3-1 and Figure 3-2, the upper part of the figures describe link options where the LP and the X710-TM4 are aligned. The bottom part of the figures describe link options where the LP and the X710-TM4 are reversed in order.

- Figure 3-3 applies when either the LP or the X710-TM4 is physically set to x8.
- Figure 3-2 applies when either the LP or the X710-TM4 is physically set to x4 and both are not physically set x8.
- Figure 3-1 applies when both the LP or the X710-TM4 is physically set to x1.

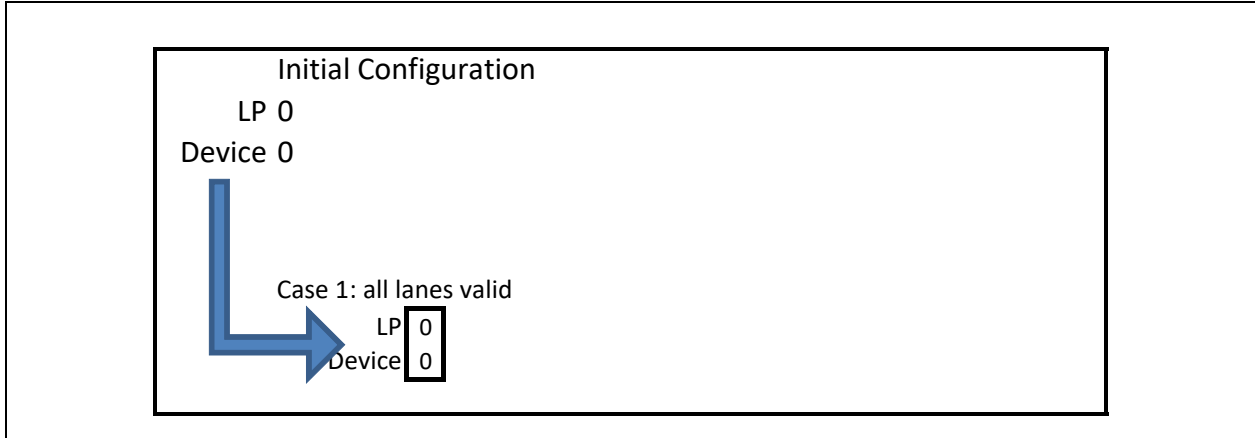


Figure 3-1. Link Width Configurations For a x1 Port

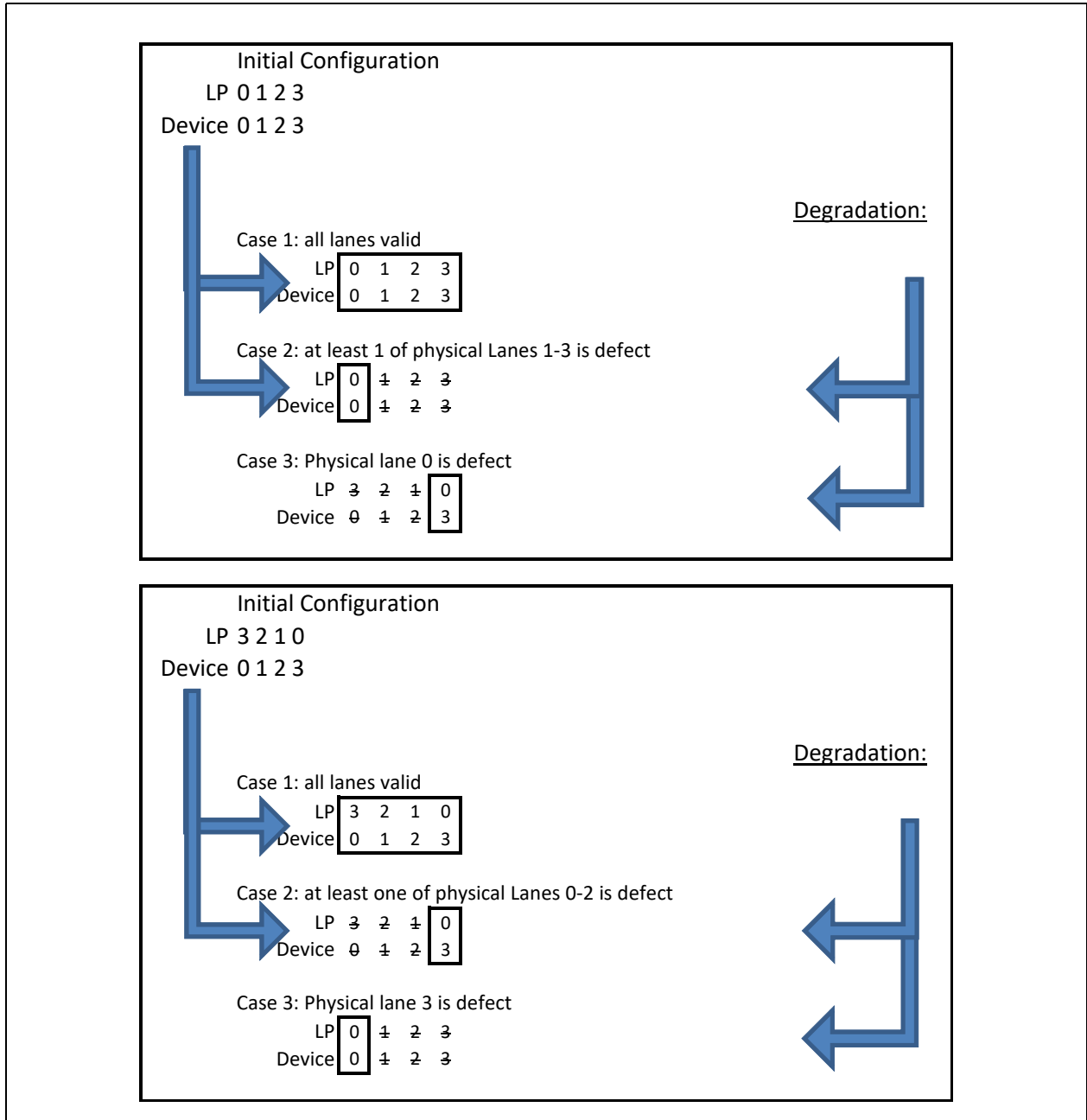
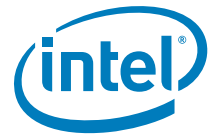


Figure 3-2. Link Width Configurations For a x4 Port

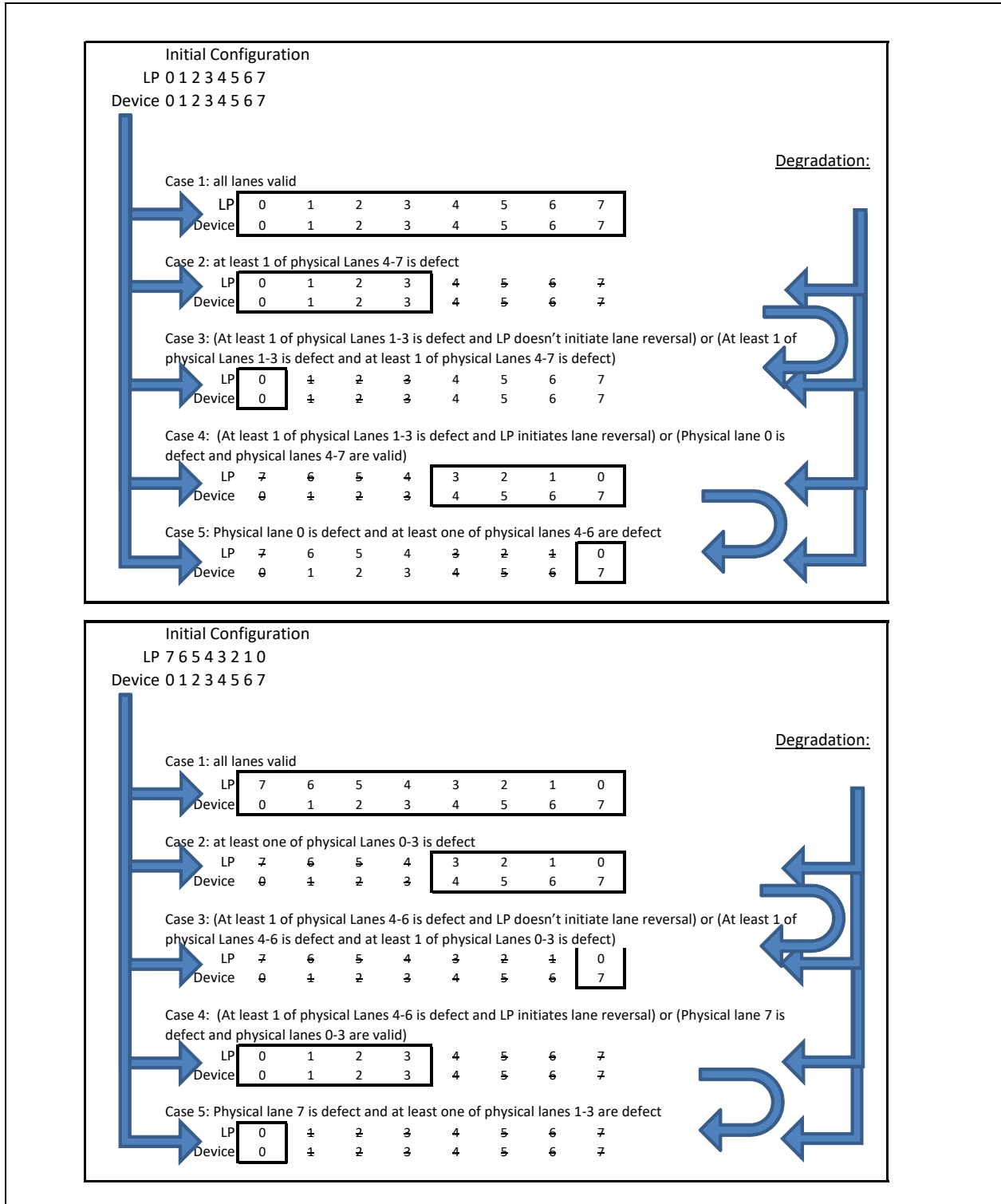


Figure 3-3. Link Width Configurations For a x8 Port



3.1.4.4 Receiver Framing Requirements

This section applies to Gen 3 operation only and lists the optional capabilities defined in section 4.2.2.3.3 (Receiver Framing Requirements) of the PCIe base specification.

The device implements the optional Gen3 receiver framing error checks other than:

- TLP Token length=0b
- Mixed order sets across lanes (anyway ending up with recovery)

3.1.5 Error Events and Error Reporting

This section describes error reporting, advanced error reporting.

3.1.5.1 General Description

PCIe defines two error reporting paradigms: the baseline capability and the Advanced Error Reporting (AER) capability. The baseline error reporting capabilities are required of all PCIe devices and define the minimum error reporting requirements. The AER capability is defined for more robust error reporting and is implemented with a specific PCIe capability structure. Both mechanisms are supported by the X710-TM4, but the AER capability needs to be enabled in the NVM.

The *SERR# Enable* and the *Parity Error* bits from the Legacy Command register also take part in the error reporting and logging mechanism.

In a multi-function device, PCIe errors that are not related to any specific function within the device are logged in the corresponding status and logging registers of all functions in that device (see Section 6.2.4 in the PCIe base specification). [Figure 3-4](#) shows, in detail, the flow of error reporting in PCIe. See also [Figure 6-2](#) in the PCIe base specification.

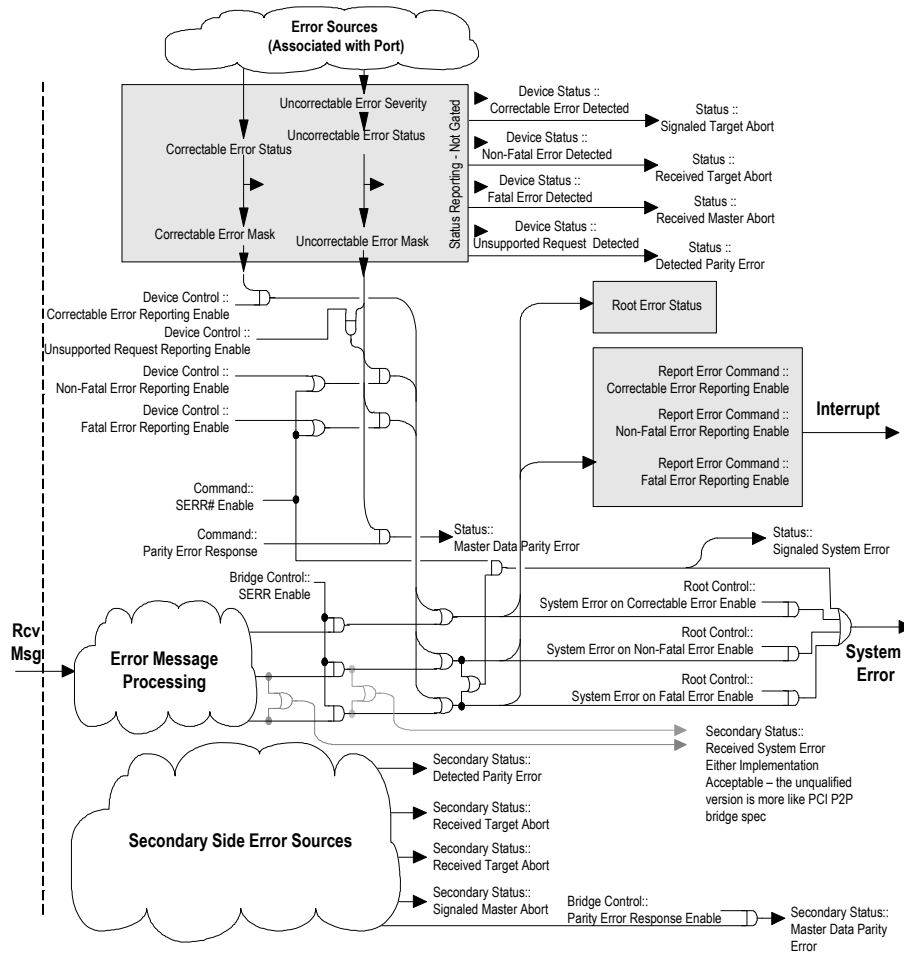


Figure 3-4. Error Reporting Mechanism

3.1.5.2 Error Events

Table 3-6 lists the error events identified by the X710-TM4 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 3-6. Response and Reporting of PCIe Error Events

Error Name	Error Events	Default Severity	Action
Physical Layer Errors			
Receiver 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 Not Legal 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 Protocol Error	Violations of Flow Control Initialization Protocol	Uncorrectable Send ERR_FATAL	
TLP Errors			
Poisoned TLP Received	TLP With Error Forwarding	Uncorrectable ERR_NONFATAL Log Header	See section Section 3.1.5.4 for more details. If error is defined as non-fatal (default severity): <ul style="list-style-type: none"> • Treat as an advisory non-fatal error: Send an ERR_COR Message If error is defined as fatal: <ul style="list-style-type: none"> • Send ERR_FATAL message
ECRC Check Failed	Failed ECRC check	Uncorrectable ERR_NONFATAL Log Header	See Section 3.1.2.9 for more details. If error is defined as non-fatal (default severity): <ul style="list-style-type: none"> • Send an ERR_NONFATAL Message If error is defined as fatal: <ul style="list-style-type: none"> • Send ERR_FATAL message
Unsupported Request (UR)	Receipt of TLP with unsupported Request Type Receipt of an Unsupported Vendor Defined Type 0 Message 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	Uncorrectable ERR_NONFATAL Log header	Send Completion With UR If error is defined as non-fatal (default severity): <ul style="list-style-type: none"> • Treat as an advisory non-fatal error: Send an ERR_COR Message If error is defined as fatal: <ul style="list-style-type: none"> • Send ERR_FATAL message
Completion Timeout	Completion Timeout Timer Expired	Uncorrectable ERR_NONFATAL	See Section 3.1.5.3 for more details. If error is defined as non-fatal (default severity): <ul style="list-style-type: none"> • Treat as an advisory non-fatal error: Send an ERR_COR Message If error is defined as fatal: Send ERR_FATAL message



Table 3-6. Response and Reporting of PCIe Error Events

Error Name	Error Events	Default Severity	Action
Completer Abort	Received Target Access with illegal data size per Section 3.1.2.2	Uncorrectable. ERR_NONFATAL Log header	Send completion with CA If error is defined as non-fatal (default severity): • Treat as an advisory non-fatal error: Send an ERR_COR Message If error is defined as fatal: • Send ERR_FATAL message
Unexpected Completion	Received Completion Without a Request For It (Tag, ID, etc.)	Uncorrectable ERR_NONFATAL Log Header	Discard TLP If error is defined as non-fatal (default severity): • Treat as an advisory non-fatal error: Send an ERR_COR Message If error is defined as fatal: • Send ERR_FATAL message
Receiver Overflow	Received TLP Beyond Allocated Credits	Uncorrectable ERR_FATAL	Receiver Behavior is Undefined
Flow Control Protocol Error	Minimum Initial Flow Control Advertisements Flow Control Update for Infinite Credit Advertisement	Uncorrectable. ERR_FATAL	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 4KB boundary	Uncorrectable ERR_FATAL Log Header	Drop the Packet, Free FC Credits
Completion with Unsuccessful Completion Status		No Action (already done by originator of completion)	Free FC Credits (no credits for completions)

3.1.5.3 Completion Timeout Mechanism

The X710-TM4 supports completion timeout as defined in the PCIe specification.

The X710-TM4 controls the following aspects of completion timeout:

- Disabling or enabling completion timeout
 - The PCIe *Completion Timeout Disable Supported* bit in the Device Capabilities 2 register is hardwired to 1b to indicate that disabling completion timeout is supported
 - The PCIe *Completion Timeout Disable* bit in Device Control 2 register controls whether completion timeout is enabled
- A programmable range of timeout values
 - The X710-TM4 supports all four ranges as programmed in the *Completion Timeout Ranges Supported* field of the Device Capabilities 2 register. The actual completion timeout value is written in the *Completion Timeout Value* field of Device Control 2 register.
- The following sequence takes place when completion timeout is detected: The appropriate message is sent on PCIe as listed in [Table 3-6](#).



- The affected queue or client takes action based on the nature of the original request:
 - If the original request was for Tx packet data, the request and any partial packet completions are dropped
 - Else, the request is handled same way as malicious requests. An interrupt is issued to the respective PF.
- Software might identify the source of the event (whether due to TLP poisoning, to a completion timeout, or an actual malicious event) by reading the error reporting counters or the performance and statistics counters.

3.1.5.4 Error Forwarding (TLP Poisoning)

If a TLP is received with an error-forwarding trailer, the packet is dropped and is not delivered to its destination.

The following sequence takes place when a poisoned TLP is received:

- The appropriate message is sent on PCIe as listed in [Table 3-6](#).
- If the TLP is a completion, a completion timeout follows at some later time. Processing continues as described in [Section 3.1.5.3](#).

System logic is expected to trigger a system-level interrupt to inform the operating system of the problem. Operating systems can then stop the process associated with the transaction, re-allocate memory to a different area instead of the faulty area, etc.

3.1.5.5 Completion With Unsuccessful Completion Status

A completion arriving with unsuccessful completion status (either UR or CA) is dropped and not delivered to its destination. A completion timeout follows at some later time. Processing continues as described in [Section 3.1.5.3](#).

3.1.5.6 Error Pollution

Error pollution can occur if error conditions for a given transaction are not isolated to the error's first occurrence. If the PHY detects and reports a receiver error, to avoid having this error propagate and cause subsequent errors at the upper layers, the same packet is not signaled at the data link or transaction layers. Similarly, when the data link layer detects an error, subsequent errors that occur for the same packet are not signaled at the transaction layer.

3.1.5.7 Blocking on Upper Address

The *GLPCI_UPADD* register blocks master accesses from being sent out on PCIe if the TLP address exceeds some upper limit. Bits [31:1] correspond to bits [63:33] in the PCIe address space, respectively.

When a bit is set in *GLPCI_UPADD*[31:1], any transaction, in which the corresponding bit in its address is set, is blocked and not sent over PCIe. If all register bits are cleared, there is no effect (for example, a packet is sent unconditionally).



Processing a blocked transaction:

- Write transaction
 - The transaction is dropped.
- Read transaction
 - The transaction is dropped.
 - Set the exceeded upper address limit (read requests) event in the PCIe errors register (see [Section 3.1.6](#)).
 - The affected queue or client takes action based on the nature of the original request. An interrupt is issued to the respective PF. See [Section 7.6.2.2.1](#).

3.1.6 Performance and Statistics Counters

The X710-TM4 incorporates counters to track the behavior and performance of the PCIe interconnect. The X710-TM4 implements several types of counters:

- Transaction layer event counters - [Section 3.1.6.1](#)
- Link and Physical layer event counters - [Section 3.1.6.2](#)
- Latency counters - [Section 3.1.6.3](#)
- Bandwidth counters - [Section 3.1.6.2.1](#)

General characteristics of the counters:

- Software can reset, stop, or start the counters.
- The counters are shared by all PCI functions (service mode of sharing).

3.1.6.1 Event Counters (Transaction Layer)

Counters operate in one of the following modes:

- Count mode — the counter increments when the respective event occurred
- Leaky bucket mode — the counter increments only when the rate of events exceeded a certain value. See [Section 3.1.6.1.2](#) for more details.

The list of events supported by the X710-TM4 are listed in [Table 3-7](#).

Table 3-7. PCIe* Statistic Events Encoding

Events	Event Mapping (Hex)	Description
Cycles	0x00	Increment on each PCIe clock tick.
Transaction Layer events		
Bad Request TLPs	0x10	Number of bad TLPs arriving to the transaction layer. These include: <ul style="list-style-type: none">• Request caused UR.• Request caused CA.• Malformed TLP.



Table 3-7. PCIe* Statistic Events Encoding

Events	Event Mapping (Hex)	Description
Bad Completions	0x11	Number of bad completions received. These include: <ul style="list-style-type: none"> • Unexpected completion. • UR status. • CA status.
Completion Timeout	0x12	Number of completion timeout events.
Poisoned TLP	0x13	Number of TLPs received with poisoned data.
ECRC Check	0x14	Number of TLPs that fail ECRC check.
Link Layer Events		
Retry Buffer Timeout	0x31	Number of replay events that happen due to timeout (does not count replay initiated due to NACK).
Retry Buffer Replay Roll-Over	0x32	Increment when a replay is initiated for more than three times.
Physical Layer Events		
Receive Error	0x50	Increment when one of the following occurs: <ol style="list-style-type: none"> 1. Decoder error occurred during training in the PHY. It is reported only when training ends. 2. Decoder error occurred during link-up or till the end of the current packet (in case the link failed). This error is masked when entering/exiting EI.

3.1.6.1.1 Count Mode

The following CSR fields control operation of the count mode:

- Four 32-bit counters `GLPCI_GSCN_0_3` track events and increment on each occurrence of an event.
 - The four 32-bit counters can also operate in a two 64-bit mode to count long intervals or large payloads.
 - Registers `GLPCI_GSCN_0_3[0]` and `GLPCI_GSCN_0_3[1]` form the first 64-bit counter. Registers `GLPCI_GSCN_0_3[2]` and `GLPCI_GSCN_0_3[3]` form the second 64-bit counter.
 - The `GLPCI_GSCL_1.GIO_64_BIT_EN` selects between 32-bit and 64-bit modes
- The `GLPCI_GSCL_1.GIO_COUNT_EN_[3:0]` bits enable each of the 4 counters
 - The enable bits for the two 64-bit counters are `GLPCI_GSCL_1.GIO_COUNT_EN_0` and `GLPCI_GSCL_1.GIO_COUNT_EN_2`, respectively.
- The `GLPCI_GSCL_1.GIO_COUNT_START` bit starts event counting of enabled counters
- The `GLPCI_GSCL_1.GIO_COUNT_STOP` bit stops event counting of running counters
- The `GLPCI_GSCL_1.GIO_COUNT_RESET` bit resets the event counters
- The `GLPCI_GSCL_2` associates an event with each of the 4 counters
 - In 64-bit mode, the `GIO_EVENT_NUM_[2,0]` fields are used

3.1.6.1.2 Leaky Bucket Mode

Each of the counters can be configured independently to operate in a leaky bucket mode. When in leaky bucket mode, the following functionality is provided:

- One of four 16-bit Leaky Bucket Counters (LBC) is enabled via the `LBC_ENABLE_[3:0]` bits in the PCIe Statistic Control register #1.



- The LBC is controlled by the *GIO_COUNT_START*, *GIO_COUNT_STOP*, *GIO_COUNT_RESET* bits in the PCIe Statistic Control register #1.
- The LBC increments every time the respective event occurs.
- The LBC is decremented every T μ s as defined in the *LBC_TIMER_N* field in the PCIe Statistic Control registers #5...#8 (*GLPCI_GSCL_5_8*).
- When an event occurs and the value of the LBC meets or exceeds the threshold defined in the *LBC_THRESHOLD_N* field in the PCIe Statistic Control registers #5...#8 (*GLPCI_GSCL_5_8*), the respective statistics counter increments, and the LBC counter is cleared to zero.

3.1.6.2 Event Counters (Link and Physical Layers)

This section describes the performance events for the link and physical layers and how to manage the counters associated with these events.

The registers responsible for the link and physical layers counters are accessed via the *GLPCI_LCBADD* and *GLPCI_LCBDATA* register pair.

Two events can be counted concurrently. The event counters include two sets of registers, each managing one event counter. Such pairs are documents as `<register_name>[1:0]`.

The following procedures manage the operation of the event counters (when writing to part of the register, make sure other fields are written with their existing values):

- Resetting the counters configuration
 - Set the *XPPERFCON.GRST* bit
 - Clear the *XPPERFCON.GRST* bit (otherwise the logic stays in reset)
- Setting an event:
 - Write 0x0...0 to the *XPPMCL[1:0]* registers
 - Set the *XPPMR[1:0].CENS* field to 0x1
 - Set the *XPPMR[1:0].CNTMD* field to 0x1
 - Set the *XPPMER[1:0].XPRSCA* field to 0x1
 - Set the event listed in [Table 3-8](#)
- Starting a count:
 - Set the *XPPERFCON.GCE* bit
- Stopping a count:
 - Clear the *XPPERFCON.GCE* bit
- Reading the count (note: reading the counter clears their values):
 - Read the respective *XPPMDH[1:0]* and *XPPMDL[1:0]* register pair in a single 64-bit aligned access.



Table 3-8 lists the link and physical layer events:

Table 3-8. Link and Physical Layer Performance Events

Event	Description	Register field
Uncorrectable Errors	Counts the total number of uncorrectable errors	XPPMER[1:0].CNTUCERR
Correctable Errors	Counts the total number of correctable errors	XPPMER[1:0].CNTCERR
Tx L0s State Utilization	Counts the number of entries to L0s on the Tx lanes	XPPMER[1:0].TXL0SU
Rx L0s State Utilization	Counts the number of entries to L0s on the Rx lanes	XPPMER[1:0].RXL0SU
Link Utilization	Counts clocks that a port is receiving data. If one counter counts receiver errors and another counter counts link utilization, a bit error rate can be calculated.	XPPMER[1:0].LNKUTIL
Recovery State Utilization	Counts the number of entries to recovery state	XPPMER[1:0].RECOVERY
ASPM L1 State Utilization	Counts the number of entries to ASPM L1 state (for example, initiated by the device)	XPPMER[1:0].L1
SW L1 State Utilization	Counts the number of entries to L1 state initiated by software	XPPMER[1:0].SWL1
Tx and Rx L0s Utilization	Counts number of events where both Tx and Rx are in L0s state	XPPMER[1:0].RXL0STXL0SU
NAK DLLP Received	Counts number of received NAK DLLPs	XPPMER[1:0].NAKDLLP

3.1.6.2.1 Bandwidth Counters

The bandwidth counters measure total payload bytes transferred over the PCIe link. Counting is provided per each traffic type (posted, non-posted, completions) per direction (upstream, downstream).

The mechanisms described above hold for the bandwidth counters with the following differences:

- Setting an event:
 - Set the XPPMR[1:0].CENS field to 0x1
 - Set the XPPMR[1:0].EGS field to 0x2
 - Set the XPPMER[1:0].FCCSEL field to the desired traffic type (posted, non-posted, completions, or all)
 - Set the XPPMER[1:0].TXRXSEL field to desired values
 - Set the XPPMER[1:0].XPRSCA field to 0x1
 - Set the XPPERFCON.GCE field to 0x1

Registers fields used exclusively by the bandwidth counters:

- XPPMER[1:0].FCCSEL - selects the desired traffic type (posted, non-posted, completions, or all)
- XPPMER[1:0].TXRXSEL - Selects between monitoring downstream traffic, upstream traffic, or both



3.1.6.2.2 Registers Map

Table 3-9. XP PM Compare Low Bits Register (XPPMCL[1:0]) (0x3288, 0x328C)

Field	Bit(s)	Init.	Description
CMPL	31:0	0xFF...F	PM compare low value. Low order bits [31:0] for PM compare register[1:0].

Table 3-10. XP PM Data Low Bits Register (XPPMDL[1:0]) (0x32E8, 0x32F0)

Field	Bit(s)	Init.	Description
CNTL	31:0	0x00...00	PM data counter low value. Low order bits [31:0] for PM data counter[1:0].

Note: XPPMDL must be read together with the respective XPPMDH register as a single 64-bit aligned read. The registers are simultaneously cleared on read.

Table 3-11. XP PM Data High Bits Register (XPPMDH[1:0]) (0x32EC, 0x32F4)

Field	Bit(s)	Init.	Description
RSVD	31:4	0x0...0	Reserved
CNTH	3:0	0x0	PM data counter high value High order bits [35:32] for PM data counter[1:0].

Note: XPPMDH must be read together with the respective XPPMDL register as a single 64-bit aligned read. The registers are simultaneously cleared on read.

**Table 3-12. XP PM Response Control Register (XPPMR[1:0]) (0x3294, 0x3298)**

Field	Bit(s)	Init.	Description
EGS	20:19	0x0	Event Group Selection
CNTMD	15:14	0x0	Count Mode
CENS	13:11	0x0	Counter enable source

Table 3-13. XP PM Resource Events Register (XPPMER[1:0]) (0x32AC, 0x32B0)

Field	Bit(s)	Init.	Description
NAKDLLP	29	0b	NAK DLLP received - set to enable
RXL0STXL0SU	28	0b	Tx and Rx L0s utilization - set to enable
SWL1	27	0b	SW L1 state utilization - set to enable
L1	26	0b	ASPM L1 state utilization - set to enable
RECOVERY	25	0b	Recovery State Utilization - set to enable
CNTUCERR	24	0b	Count Uncorrectable Errors - set to enable
CNTCERR	23	0b	Count Correctable Errors - set to enable
TXL0SU	22	0b	Tx L0s state utilization event - set to enable
RXL0SU	21	0b	Rx L0s state utilization event - set to enable
XPRSCA	20:17	0x0	XP Resource Assignment 0001b: Set Else: reserved
LNKUTIL	16:13	0x0	Link Utilization - set to 0x1 to enable
FCCSEL	4:2	0x0	Flow Control Class Select This field selects which flow class for resource event xx1: Posted x1x: Non-Posted 1xx: Completion Note: setting to 111b counts posted, non-posted and completion traffic combined
TXRXSEL	1:0	0x0	Tx/Rx Select (TXRXSEL): This field selects the traffic direction to monitor: 1x: Transmit x1: Receive (from PCIe bus) 11: Either Transmit or Receive direction

Table 3-14. Performance Monitor Local Control Register (XPPERFCON) (0x32C4)

Field	Bit(s)	Init.	Description
RSVD	31:2	0x0	Reserved
GCE	1	0b	Global Count Enable
GRST	0	0b	Global Reset



3.1.6.3 Latency Counter

The latency counter measures the min, max, or average read latency.

Note: Completion timeout events are ignored when the latency counter is enabled.

The latency counters are managed via a set of register fields described below (see also [Table 3-15](#), [Table 3-16](#), and [Table 3-17](#)). Each of the following sources of traffic has its separate set of registers and counters:

- 0x0 - Rx LAN descriptor fetch
- 0x1 - Tx LAN descriptor fetch
- 0x4 - Internal cache load
- 0x5 - Internal management engine read
- 0x6 - Tx LAN packet fetch

The registers are accessed via the GLPCI_LCBADD and GLPCI_LCBDATA registers.

The register fields that control the latency counter operation are:

**Table 3-15. NPQ Control Register - NPQC (0x00000)**

Field	Bit(s)	Init.	Description
Reserved	31:10	0	Reserved
RTMNTREN	9	0	Latency counting enable. This bit should set in order to enable latency counters. When set, completion timeout events are ignored
PERFMNTREN	8	0	Performance monitor enable. This bit should set in order to enable latency counters. Clearing this bit clears the latency counters
PERFMNTRAVG	7:4	1	Performance monitor average rate. This field sets the averaging rate for all latency average monitors. See definition of NPQRTDLY1.ARTDLY This field divided by 16 is the weight W in an exponential moving average. The possible values are 1, 2, 4 or 8, which correspond to averaging rates of 0.0625, 0.125, 0.25 or 0.5, resp.
Reserved	3:0	0x4	Reserved

Table 3-16. NPQ Round-trip Delay 1 Register - NPQRTDLY1 (0x00030; RO)

Field	Bit(s)	Init.	Description
Reserved	31:16	0	Reserved
ARTDLY	15:0	0	Average read requests round-trip delay. Captures the average read latency experienced since the last counter reset. Latency is measured from time the read request starts until the time the completion starts to arrive Average is calculated as exponential moving average. That is, the new average $M(n)$ at sample n equals $M(n) = (W/16) * \text{new_sample} + (16-W)/16 * M(n-1)$, where W is defined in the NPQC.PERFMNTRAVG field

Table 3-17. NPQ Round-trip Delay 2 Register - NPQRTDLY2 (0x00034; RO)

Field	Bit(s)	Init.	Description
MAXRTDLY	31:16	0	Maximal read requests round-trip delay. Captures the maximal read latency experienced since the last counter reset. Latency is measured from time the read request starts until the time the completion starts to arrive
MINRTDLY	15:0	0	Minimal read requests round-trip delay. Captures the minimal read latency experienced since the last counter reset. Latency is measured from time the read request starts until the time the completion starts to arrive

Latencies are measured in cycle counts, where a cycle duration is listed in [Table 3-18](#).



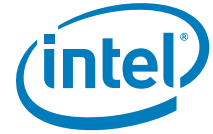
Table 3-18. Resolution of the Latency Counters

PCIe Operation Speed	Setting of the GLPCI_CLKCTL.PCI_CLK_DYN Bit	PCIe Operational Link Width	Cycle Duration (ns)
Gen 1 (2.5G)	0b	x	8
Gen 2 (5.0G)	0b	x	4
Gen 3 (8.0G)	0b	x	2
Gen 1 (2.5G)	1b	8 lanes	16
Gen 2 (5.0G)	1b	8 lanes	8
Gen 3 (8.0G)	1b	8 lanes	4
Gen 1 (2.5G)	1b	1 or 4 lanes	32
Gen 2 (5.0G)	1b	1 or 4 lanes	16
Gen 3 (8.0G)	1b	1 or 4 lanes	8

The NPQC register serves other purposes, so using it should be done in the following manner:

- Read the relevant NPQC register
- Modify the relevant fields from the list above in a local copy
- Write the local copy back to the device

Clearing the latency counters is done by setting the NPQC.PERFMNTREN bit to 0b.



3.2 Ethernet Interconnect

3.2.1 Media Access Control (MAC) Layer

The X710-TM4 supports up to four full duplex Ethernet MAC ports compliant with IEEE Std 802.3-2012 standard (Clause 4 and Annex 4A). The MAC ports can be configured to operate at different speeds of operation as described in [Section 3.2.1.1](#).

Each MAC port is associated with a corresponding Media Access Unit (MAU) that provides the physical layer interfaces. The MAUs need to be configured to operate with appropriate physical layer protocols based on the MAC operating speed. Physical layers supported by the X710-TM4 for different speeds of operation are explained in [Section 3.2.7](#).

3.2.1.1 MAC Speed Configuration

[Table 3-19](#) lists the possible speed configurations available on each MAC port.

Table 3-19. MAC Port and Possible Speed Configurations

Port	100 Mb/s	1 Gb/s	10 Gb/s
MAC 0	Y	Y	Y
MAC 1	Y	Y	Y
MAC 2	Y	Y	Y
MAC 3	Y	Y	Y

Each port marked as “Y” in the previous table might be enabled for the relevant speed. As listed in the previous table, in some setups only some of the ports might be enabled.

When LAN ports are disabled in multi-port system configurations, corresponding PCIe functions need to be disabled through the NVM or external disable pins. See [Section 4.5](#) for details on PCI function disable and LAN port disable functionality.

The link speed for each port can be controlled through NVM loaded settings or Link Configuration Admin commands (see [Section 3.2.12](#)). In some physical interfaces the final link speed is selected, out of the pre-configured options, based on the link auto-negotiation protocol.

3.2.1.2 Ethernet CRC Generation and Stripping

The X710-TM4 MAC supports CRC generation (Ethernet frame check sequence). The MACs can be independently programmed to generate and append the 32-bit CRC (FCS) in the transmit direction by setting the *CRC Enable* field in the Set MAC Config AQ command.

The default value is set for the MAC to append the CRC.



Note: When CRC generation is disabled in the MAC, automatic padding in the MAC does not work correctly, therefore all transmitted packets must be at least 64 bytes long.

3.2.1.3 Transmit Padding

The minimum frame size for Ethernet as specified by IEEE Std 802.3 standard is 64 bytes. The X710-TM4 MAC pads, with zeros, Ethernet packets that are smaller than 64 bytes during transmit. Refer to the padding rules for received packets and loop-back packets in [Section 7.2.2](#).

3.2.1.4 Jumbo Frame Support

The X710-TM4 MACs support transmitting and receiving frames of up to 9.5 KB (9728 bytes). Maximum receive and transmit frame size is configured through the Set MAC Config <Max Frame Size > field.

3.2.1.5 Ethernet Flow Control (FC)

The X710-TM4 supports flow control (pause) as defined in 802.3x (IEEE Std 802.3-2008 Annex 31B), as well as the specific operation of asymmetrical flow control (asymmetric pause) defined by 802.3z (IEEE Std 802.3-2008 Annex 28B). The X710-TM4 also supports Priority Flow Control (PFC) as defined in IEEE P802.1Qbb, sometimes referred to as Class Based Flow Control or (CBFC), as part of the DCB architecture.

Note: A X710-TM4 port can either be configured to receive 802.3x Link Flow Control (LFC) packets or 802.1Qbb/802.3bd PFC packets. It does not support the reception of both types of packets simultaneously over the same port.

Note: LFC/PFC is not supported at 2.5 Gb/s or 5 Gb/s speeds.

Flow control is implemented to reduce receive buffer overflows, which result in the dropping of received packets. Flow control also allows for local controlling of network congestion levels. This can be accomplished by sending an indication to a transmitting station of a nearly full receive buffer condition at a receiving station.

The implementation of asymmetric flow control allows for one link partner to send flow control packets while being allowed to ignore their reception (for example, not required to respond to PAUSE frames).

The following registers define the basic control functionality. Refer to the registers specified in [Section 7.7.1.2.10](#) and [Section 7.7.2.2](#) for the other programming related to flow control. In DCB mode, some of the registers are duplicated per Traffic Class (TC), up to eight duplicate copies of the registers. If DCB is disabled, index [0] of each register is used.

- MAC Flow Control (PRTDCB_MFLCN) register — Enables flow control and passing of MAC control packets to the host.
- Flow Control Configuration (PRTDCB_FCCFG) — Determines mode for Tx flow control (no FC, LFC, or versus PFC) Note that if Tx flow control is enabled then Tx CRC by hardware should be enabled as well (Set MAC Config <CRC Enable> = 1b).
- Flow Control Transmit Timer Value (PRTDCB_FCTTVN[3:0]) — a set of 16-bit timer values to include in transmitted PAUSE frame. A single timer is used in LFC mode and up to eight timers are used in PFC mode.
- Flow Control Refresh Threshold Value (PRTDCB_FCRTV) — 16-bit PAUSE refresh threshold value (in legacy FC PRTDCB_FCRTV must be smaller than PRTDCB_FCTTVN[0]).



3.2.1.5.1 MAC Control Frames and Receiving Flow Control Frames

3.2.1.5.1.1 MAC Control Frame — Other Than FC

IEEE 802 reserved the Ethertype value of 0x8808 for MAC control frames as listed in [Table 3-20](#). The MAC control frame format is specified in IEEE Std. 802.3-2018 Clause 31.

Table 3-20. MAC Control Frame Format

DA	The <i>Destination Address</i> field can be an individual or multicast (including broadcast) address. Permitted values for the <i>Destination Address</i> field can be specified separately for a specific control opcode such as FC packets.
SA	Port Ethernet MAC address (6 bytes).
Type	0x8808 (2 bytes).
Opcode	The MAC control opcode indicates the MAC control function.
Parameters	The <i>MAC Control Parameters</i> field must contain MAC control opcode-specific parameters. This field can contain none, one, or more parameters up to a maximum of <code>minFrameSize = 20</code> bytes.
Reserved field = 0x00	The <i>Reserved</i> field is used when the MAC control parameters do not fill the fixed length MAC control frame.
CRC	4 bytes.

3.2.1.5.1.1.1 MAC Control Frame Receive Identification

Packets with:

- MAC DA = 01-80-C2-00-00-01.
- Ethertype value of 0x8808 are considered to be control frames that follow.

The 10 Gb/s and lower speeds MAC supports the following configurations:

- UC MAC DA (PRTGL_SAL and PRTGL_SAH).

3.2.1.5.1.2 Structure of 802.3x FC Packets

802.3x FC packets are defined by the following three fields (see [Table 3-21](#)):

1. A match on the six-byte multicast address for MAC control frames or a match to the station address of the device. The 802.3x standard defines the MAC control frame multicast address as 01-80-C2-00-00-01. A match on the *Type* field. The *Type* field in the FC packet is compared against an IEEE reserved value of 0x8808.
2. A match of the *MAC Control Opcode* field has a value of 0x0001.

Frame-based flow control differentiates XOFF from XON based on the value of the *PAUSE Timer* field. Non-zero values constitute XOFF frames while a value of zero constitutes an XON frame. Values in the *Timer* field are in units of pause quanta (slot time). A pause quanta lasts 64 byte times, which is converted in to an absolute time duration according to the line speed.

Note: XON frame signals the cancellation of the pause that was initiated by an XOFF frame. For example, a pause for zero pause quanta.

**Table 3-21. 802.3x Link Flow Control - Frame Formats**

DA	01_80_C2_00_00_01 (6 bytes).
SA	Port Ethernet MAC address (6 bytes).
Type	0x8808 (2 bytes).
Opcode	0x0001 (2 bytes).
Time	XXXX (2 bytes).
Pad	42 bytes.
CRC	4 bytes.

3.2.1.5.1.2.1 802.3x Frame Receive Identification

Received frames that are identified as control frames (see [Section 3.2.1.5.1.1.1](#)) can further be classified as 802.3x if the frames opcode = 0x0001 as listed in [Table 3-21](#).

The 10 Gb/s and lower speeds MAC supports the following configuration to further manage 802.3x FC frames' Rx identification:

- UC MAC DA (PRTGL_SAL and PRTGL_SAH).

3.2.1.5.1.3 Priority Flow Control (PFC)

DCB introduces support for multiple TCs assigning different priorities and bandwidth per TC. LFC stops all the TCs. PFC specified in IEEE P802.1Qbb enables more granular flow control on the Ethernet link in a DCB environment as opposed to the PAUSE mechanism defined in 802.3x. The PFC frame format is specified in IEEE P802.3bd.

Table 3-22. Priority Flow Control - Packet Format

DA	01_80_C2_00_00_01 (6 bytes).
SA	Port Ethernet MAC address (6 bytes).
Type	0x8808 (2 bytes).
Opcode	0x0101 (2 bytes).
Priority Enable Vector	0x00XX (2 bytes).
Timer 0	XXXX (2 bytes).
Timer 1	XXXX (2 bytes).
Timer 2	XXXX (2 bytes).
Timer 3	XXXX (2 bytes).
Timer 4	XXXX (2 bytes).
Timer 5	XXXX (2 bytes).
Timer 6	XXXX (2 bytes).
Timer 7	XXXX (2 bytes).
Pad	26 bytes.
CRC	4 bytes.

**Table 3-23. Format of Priority Enable Vector**

	ms octet	ls octet
Priority enable vector definition	0	e[7]...e[n]...e[0]
e[n] =1 => time (n) valid e[n] =0 => time (n) invalid		

Each of the eight timers refers to a specific User Priority (UP). For example, Timer 0 refers to UP 0, etc. The X710-TM4 binds a UP (and therefore the timer) to one of its TCs according to the UP-to-TC binding tables. Refer to the PRTDCB_TUP2TC register for the binding of received PFC frames to Tx TCs, and to the PRTDCB_RUP2TC register for the binding of transmitted PFC frames to Rx TCs.

When a PFC frame is formatted by the X710-TM4, the same values are replicated into every *Timer* field and priority enable vector bit of all the UPs bound to the concerned TC. These values as configured in the PRTDCB_RUP2TC register.

The following rule is applicable for the case of multiple UPs that share the same TC as configured in the PRTDCB_TUP2TC register. When PFC frames are received with different timer values for the previous UPs, the traffic on the associated TC must be paused by the highest XOFF timer's value.

3.2.1.5.1.3.1 PFC Frame Receive Identification

Received frames that are identified as control frames (see [Section 3.2.1.5.1.1.1](#)) can further be classified as PFC if the frames opcode = 0x0101 as listed in [Table 3-22](#).

The 10 Gb/s and lower speeds MAC supports the following configurations to manage PFC frames' identification on Rx:

- UC MAC DA (PRTGL_SAL and PRTGL_SAH).

3.2.1.5.1.4 Operation and Rules

The X710-TM4 operates in either LFC mode or in PFC mode. Enabling both modes concurrently is not allowed.

The 10 Gb/s and lower speeds MAC implements the following configurations:

- LFC (PAUSE) is enabled by the *RFCE* bit in the PRTDCB_MFLCN register.
- PFC is enabled by the *RPFCE* bit in the PRTDCB_MFLCN register.

Note: LFC capability must be negotiated between link partners via the auto-negotiation process. PFC capability is negotiated via some higher level protocol (DCBX) and the resolution is usually provided to the software device driver by the DCB management agent. It is the EMP responsibility to reconfigure the LFC settings after the auto-negotiation process was resolved.

Once the receiver has validated the reception of an XOFF, or PAUSE frame, the device performs the following:

- Increments the appropriate statistics register(s).
- Initialize the pause timer based on the packet's PAUSE *Timer* field (overwriting any current timer's value).
 - In case of PFC, this is done per TC. If several UPs are associated with a TC, then the device sets the timer to the maximum value among all enabled timer fields associated with the TC.



- Disable packet transmission or schedule the disabling of transmission after the current packet completes.
 - In case of PFC, this is done per paused TC.

Resuming transmission can occur under the following conditions:

- Expiration of the PAUSE timer.
 - In case of PFC, this is done per TC.
- Receiving an XON frame (a frame with its PAUSE timer set to 0b).
 - In case of PFC, this is done per TC.

Both conditions clear the relevant TXOFF status bits in the Transmit Flow Control Status (PRTDCB_TFCS) register and transmission can resume. Hardware records the number of received XON frames.

3.2.1.5.1.5 Timing Considerations

When operating at 10 Gb/s line speed, the X710-TM4 must not begin to transmit a (new) frame more than 60 pause quanta after receiving a valid XOFF frame, as measured at the wires. When connected to an external 10GBASE-KR PHY with FEC or to an external 10GBASE-T PHY, the response time requirement increases to 74 pause quanta, because of extra delays consumed by these external PHYs.

When operating at 1 Gb/s line speed, the X710-TM4 must not begin to transmit a (new) frame more than 2 pause quanta after receiving a valid XOFF frame, as measured at the wires.

The IEEE P802.1Qbb draft 2.3, specifies that the tolerated response time for priority XOFF frames is 614.4 ns (equivalent of 12 pause quanta at the link speed of 10 Gb/s. This extra budget in addition to the link delay is aimed to compensate for the fact that a decision to stop new transmissions from a specific TC must be taken earlier in the transmit data path than for the LFC case.

3.2.1.5.2 PAUSE and MAC Control Frames Forwarding

3.2.1.5.2.1 At 10 Gb/s and Lower Speeds

Two bits in the PRTDCB_MFLCN register control the transfer of PAUSE and MAC control frames to the host. These bits are Discard PAUSE Frames (DPF) and Pass MAC Control Frames (PMCF). Note also that any packet must pass the L2 filters as well.

- The *DPF* bit controls transfer of PAUSE packets to the host. The same policy applies to both LFC and PFC packets as listed in [Table 3-24](#). Note that any packet must pass the L2 filters as well.
- The *PMCF* bit controls transfer of non-PAUSE packets to the host. Note that when LFC frames are not enabled (RFCE = 0b) then LFC frames are considered as MAC Control (MC) frames for this matter. Similarly, when PFC frames are not enabled (RPFCE = 0b) then PFC frames are considered as MC frames as well.

Note: When virtualization is enabled, forwarded control packets are queued according to the regular switching procedure.

**Table 3-24. Transfer of PAUSE Packet to Host (DPF bit) in 10 Gb/s MAC**

RFCE	RPFCM	DPF	LFC Handling	PFC Handling
0b	0b	X	Treat as MC (according to PMCF setting).	Treat as MC (according to PMCF setting).
1b	0b	0b	Accept.	Treat as MC (according to PMCF setting).
1b	0b	1b	Reject.	Treat as MC (according to PMCF setting).
0b	1b	0b	Treat as MC (according to PMCF setting).	Accept.
0b	1b	1b	Treat as MC (according to PMCF setting).	Reject.
1b	1b	X	Unsupported setting.	Unsupported setting.

3.2.1.5.3 Transmitting PAUSE Frames

The X710-TM4 generates PAUSE packets to insure there is enough space in its receive packet buffers to avoid packet drop. The X710-TM4 monitors the fullness of its receive FIFOs and compares it with the contents of a programmable threshold. When the threshold is reached, the X710-TM4 sends a PAUSE frame. The X710-TM4 supports both LFC and PFC — but not both concurrently (at the same physical port). When DCB is enabled, it sends only PFC, and when DCB is disabled, it sends only LFC.

Note: Similar to receiving flow control packets previously mentioned, software can enable flow control transmission by setting the PRTDCB_FCCFG.TFCE field.

3.2.1.5.3.1 PFC

Like Rx flow control, Tx flow control operates in either a link 802.3x compliant mode or in PFC mode, but not in both at the same time.

The same flow control mechanism is used for PFC and for 802.3x flow control to determine when to send XOFF and XON packets. When PFC is used in the receive path, Priority PAUSE packets are sent instead of 802.3x PAUSE packets. The format of priority PAUSE packets is described in [Section 3.2.1.5.1.2.1](#).

A specific consideration for generating PFC packets:

- When a PFC packet is sent, the packet sets all the UPs that are associated with the relevant TC (UP-to-TC association in receive is defined in PRTDCB_RUP2TC register).

3.2.1.5.3.2 Operation and Rules

At 10 Gb/s and lower speeds, The *TFCE* field in the Flow Control Configuration (PRTDCB_FCCFG) register enables transmission of PAUSE packets as well as selects between the LFC mode and the PFC mode.

Refer to [Section 7.7.1.2.4](#) for the criteria used by the device for sending a XOFF frame to the neighbor. The X710-TM4 sends an additional PAUSE frame if it has previously sent one and the FIFO overflows. This is intended to minimize the amount of packets dropped if the first PAUSE frame did not reach its target.

From the time it has issued a PAUSE frame to the neighbor, the X710-TM4 starts counting down in an internal shadow counter that is used to mirror the pause time-out counter at the partner's end. When this internal counter reaches the value set in the iPRTDCB_FCRTV register then, if the PAUSE condition is still valid (meaning that the buffer(s) fullness is still above the relevant watermarks), an XOFF message is sent again.



Once the receive buffer fullness reaches the relevant low watermarks, the X710-TM4 sends an XON message (a PAUSE frame with a timer value of zero). Refer to [Section 7.7.1.2.7](#) for the criteria used by the device for sending a XON frame to the neighbor.

3.2.1.6 Inter Packet Gap (IPG) Control and Pacing

The X710-TM4 supports transmission pacing by extending the IPG (the gap between consecutive packets). The pacing mode enables the average data rate to be slowed in systems that cannot support the full link rate (10 Gb/s, 5 Gb/s, 2.5 Gb/s, 1 Gb/s or 100 Mb/s). As listed in [Table 3-25](#), the pacing modes work by stretching the IPG in proportion to the data sent. In this case, the data sent is measured from the end of preamble to the last byte of the packet. No allowance is made for the preamble or default IPG when using pacing mode.

Example 1:

Consider an example of a 64-byte frame. To achieve a 1 Gb/s data rate when link rate is 10 Gb/s and packet length is 64 bytes (16 Dwords), programmers need to add an additional IPG of 144 Dwords (nine times the packet size to reach 1 Gb/s). When added to the default IPG, this gives an IPG of 147 Dwords.

Example 2:

Consider an example of a 65-byte frame. To achieve a 1 Gb/s data rate when link rate is 10 Gb/s and packet length is 65 bytes (17 Dwords when rounded up) programmers need to add an additional IPG of 153 Dwords (nine times the packet duration in Dwords). When added to the default IPG, this gives an IPG of 156 Dwords. Note that in these cases, where the packet length counted in Dwords is not an integer, programmers need to count any fraction of a Dword as an entire Dword for computing the additional IPG.

[Table 3-25](#) lists the pacing configurations supported by the X710-TM4 at link rates of 10 Gb/s. When operating at lower link speeds the pacing speed is proportional to the link speed. Pacing is configured in the *Pacing Config* field in the Set MAC Config admin command.

Table 3-25. Pacing Speeds at 10 Gb/s Link Speed

Pacing Speeds (Gb/s)	Delay Inserted into IPG	Pacing Config Field in the Set MAC Config Admin Command
10 (LAN)	None	0000b
9.294196 (WAN)	1 byte for 13 transmitted	1111b
9.0	1 Dword for 9 transmitted	1001b
8.0	1 Dword for 4 transmitted	1000b
7.0	3 Dwords for 7 transmitted	0111b
6.0	2 Dwords for 3 transmitted	0110b
5.0	1 Dwords for 1 transmitted	0101b
4.0	3 Dwords for 2 transmitted	0100b
3.0	7 Dwords for 3 transmitted	0011b
2.0	4 Dwords for 1 transmitted	0010b
1.0	9 Dwords for 1 transmitted	0001b
10	None	Default



3.2.1.7 MAC Speed Change at Different Power Modes

Auto-negotiation enables establishing link speed at the Highest Common Denominator (HCD). During certain low power modes, power saving might be more important than link performance. The X710-TM4 supports an additional mode of operation, where it can configure the PHY to a fixed speed, starting from the lowest speed and checking if the link can be up. This method sets the link speed to the Lowest Common Denominator (LCD) link speed. The link-up process enables the link to come up at any possible speed in cases where power is more important than performance. See further information in [Section 5.3.2](#).

3.2.1.8 MAC Errors

The X710-TM4's MAC supports identifying the following erroneous packets:

- L2 CRC Error
Packet's FCS check resulted in an error.
- Undersized or oversized packets
Received frame size is smaller than 64 bytes or larger than the configured max frame size, which was either loaded from the NVM or set using the Set MAC Config command.
- Illegal Byte Error
Indication that an illegal control byte was received on the XGMII interface.
An illegal control byte is any value that is not legal. (See IEEE Std. 802.3-2018 for legal symbols, such as /I/, /E/, /T/, /D/, /S/, /Seq/, or /LPI/).
- Error Byte Error
Indication that a packet was received with the error control byte (/E/) on the XGMII interface. Indicates that the PCS has encountered an error during the packet's reception.
- 802.3 Length Error
<Length> field information in the 802.3 header does not match the packet's actual size.

Packets containing any one of the previous errors can be filtered by the device or forwarded to a pre-configured VSI based on the SBP flag in the PRT_SBPVSI register.

Note: Packets shorter than 64 bytes are always filtered by the MAC layer and are not affected by the Store Bad Packets (SBP) configuration.

3.2.1.8.1 MAC Error Counters

Table 3-26 lists the different MAC error counters supported by the device.

Table 3-26. MAC Error Counters

Counter Name	Description
CRC Error	Counts the number of packets received with CRC errors that are not fragments.
Illegal Byte Error	Counts the number of packets received with an illegal control byte on the XGMII interface.
Error Byte	Counts the number of packets that were received with an error control byte on the XGMII interface.

**Table 3-26. MAC Error Counters**

Counter Name	Description
Receive Length Error	Counts the number of packets received with error in length field comparison in the 802.3 header.
Receive Undersize	Counts the number of packets with good CRC that are smaller than 64 bytes.
Receive Fragment	Counts the number of packets with bad CRC that are smaller than 64 bytes.
Receive Oversize	Counts the number of packet that are larger than the configured max frame size.
MAC Short Packet Discard	Counts the number of packets smaller than 32 bytes that were discarded by the MAC layer.

3.2.2 Internal PHY for 10GBASE-T Ports

The internal PHY is a 10GBASE-T/5GBASE-T/2.5GBASE-T/1000BASE-T/100BASE-TX transceiver that performs all the Physical Layer (PHY) interface functions for 10GBASE-T, 5GBASE-T, 2.5GBASE-T, 1000BASE-T, and 100BASE-T Ethernet on Category 6, 6A, or 7 twisted-pair cable. 5GBASE-T, 2.5GBASE-T, 1000BASE-T, and 100BASE-TX operations are supported on Category 5e UTP. The PHY connects to a Media Access Controller (MAC) or switch controller through XFI/10GBASE-KR or USXGMII for 10GBASE-T, USXGMII, 5000BASE-X or 5G rate over XFI for 5GBASE-T USXGMII, 2500BASE-X or 2.5 Gb/s rate over XFI for 2.5GBASE-T and USXGMII, 1000BASE-X(SGMII) for 1000BASE-T/100BASE-TX.

The PHY is fully compliant with IEEE Std. 802.3-2018 standard for 10GBASE-T, 1000BASE-T, and 100BASE-T. Operation at 5GBASE-T and 2.5GBASE-T complies with the IEEE Std. 802.3bz standard. It connects to network twisted-pair wiring through isolation transformers. The PHY can be programmed to auto-negotiate its operating speed based on the capabilities advertised by the link partner.

3.2.2.1 10GBASE-T

The PHY supports the 10GBASE-T transmission rate, per the IEEE P802.3an standard.

3.2.2.1.1 Transmit Path

The transmitter path collects the 64-bit parallel data stream at the XGMII interface and reformats the data into 10GBASE-T/1000BASE-T over four twisted pairs.

3.2.2.1.2 Scramble and Frame

Scrambling or randomizing the transmit data is necessary to minimize baseline wander and ensure proper operation of adaptive receiver subsystems, such as clock recovery and equalization. All data entering the PHY is scrambled by a 58-bit self-synchronizing scrambler. With continuous 10GBASE-T operation, this scrambler does not repeat more often than once every 55 years.

PHY-level frames are not visible to higher layers, but are necessary to allow for proper synchronization and alignment of Low-Density Parity-Check (LDPC) blocks between the transmitter and receiver. PHY level frames contain data from 50 consecutive 65-bit XGMII data/control blocks. A CRC-8 is computed across $50 \times 65 = 3250$ bits and appended to the frame. One additional bit is added for vendor-specific purposes, bringing the total PHY-level frame payload to $(1 + 50) \times (65 + 8) = 3259$ bits.



3.2.2.1.3 LDPC Encode

A low-density parity check code is an error correcting code, a method of transmitting a message over a noisy transmission channel. The PHY adds 325 parity bits to blocks of 1723 data bits. This block encoding operation is equivalent to a matrix multiplication of $c = xG$, where x is a 1723-bit input vector $[x_0, x_1, \dots, x_{1772}]$, c is a 2048 output code vector $[c_0, c_1, \dots, c_{2043}]$, and G is the code generator matrix.

From the PHY frame payload of 3259 bits, 1723 bits are encoded by the LDPC encoder and 1536 bits remain uncoded. The 325 parity bits are added to the 3259-bit payload for a total encoded PHY frame size of 3584 bits. Code block boundaries are recovered at the receiver through PHY frame boundaries.

3.2.2.1.4 DSQ128 Map

The 3584-bit encoded PHY frame contains $3584/7 = 512$, 7-bit DSQ128 symbols. Each 7-bit symbol maps to a single point in a two-dimensional symbol constellation. Each DSQ128 symbol is transmitted in two consecutive sample intervals: first the I value, then the Q value. The I to Q ordering is aligned with the PHY frame boundary so that it can be recovered by the receiver. Each 512 symbol PHY frame is divided across four-wire pairs. In each symbol interval, four symbols are transmitted in parallel across the four wires: either four I values or four Q values.

3.2.2.1.5 PAM16 Mapping (IEEE Std. 802.3bz)

The 2048-bit encoded PHY frame contains $2048/4 = 512$, 4-bit gray-coded PAM16 symbols. Each 4-bit symbol maps to a single point in a one-dimensional symbol constellation. The frame structure for 2.5GBASE-T and 5GBASE-T modes are identical except the symbol duration is 320 ns and 640 ns for 5GBASE-T and 2.5GBASE-T, respectively. Each 512 symbol PHY frame is divided across four-wire pairs. In each symbol interval, four PAM16 symbols are transmitted in parallel across the four wires.

3.2.2.1.6 TH Precode

Each dimension (four I values or four Q values) of the four channel symbols from the 4D DSQ128 mapper is passed through a Tomlinson-Harashima Precoder (THP). The THP is similar in structure to a Decision-Feedback Equalizer (DFE), but it is used in the transmitter rather than the receiver to avoid DFE error propagation issues. The coefficient values for the THP are determined as part of the link start-up sequence and remain fixed during data transfer.

3.2.2.1.7 Digital-to-Analog Converters

Each of the four outputs from the precoders are converted to an analog signal using integrated digital-to-analog converters (DAC). The DAC drives the 100Ω twisted-pair cable (through a transformer) with a differential signal. The average transmit power is 4.2 dBm for 100 m (or greater) cables and backs off in value as the cable length is reduced.

3.2.2.1.8 Receive Path

The receiver path accepts 10GBASE-T framing over four twisted pairs and reformats the data into a 64-bit parallel data stream at the XGMII interface.



3.2.2.1.9 Four-Channel Analog-to-Digital Converters

The signal received on each of the four wire pairs is converted to a digital representation by integrated analog-to-digital converters (ADC).

3.2.2.1.10 Clock Recovery

The optimal sampling instant for each of the four ADCs is determined by the clock recovery circuit. In slave mode, the frequency and phase are derived from the received data so that the slave is frequency- and phase-locked with the master.

3.2.2.1.11 Matrix Crosstalk Canceler and Equalizer

The major signal impairments that can be canceled/suppressed are crosstalk within and between the four-wire pairs and Inter-symbol Interference (ISI) caused by the insertion loss of a wire pair. Each receiver processes crosstalk from four local transmitters: three transmitters couple through near-end crosstalk (Next) channels and one transmitter (on the same wire) couple through the echo channel (echo is caused by the non-ideal return loss of the various components of the channel). The four crosstalk sources per each of the four receivers result in a 4x4 matrix crosstalk channel connecting the four local transmitters to the four local receivers. Adaptive matrix crosstalk cancellation is used to remove the crosstalk from the signal. The matrix crosstalk canceler continuously adapts to track changes in the crosstalk and echo channels.

Each local receiver is connected to a remote transmitter by a direct channel (the wire) and to three other remote transmitters by crosstalk channels. These are also called Far-end Crosstalk (FEXT) channels. This arrangement operates as a 4x4 matrix channel connecting the four local receivers to the four remote transmitters. Adaptive matrix equalization is used to counter the effects of the matrix signal channel. The matrix equalizer continuously adapts to track changes in the FEXT and ISI channels.

3.2.2.1.11.1 Echo Canceler

Because of simultaneous bidirectional transmission of data on each cable pair in 10G/5G/2.5GBASE-T, echo impairments from the local transmitters are present at each receiver. A digital echo canceler is used to subtract an estimate of the echo from the received signal. The digital echo canceler coefficients are adaptive to track the varying echo transfer function caused by different channels, transmitters, cable geometry, and environmental conditions.

3.2.2.1.12 Four-Channel DSQ128 Soft LDP Decision and LDPC Decoder (10GBASE-T)

The objective of matrix equalization, matrix crosstalk cancellation, and clock recovery processes is to reconstruct, as closely as possible, the original DSQ128 symbol transmitted by the link partner. The LDPC decoding algorithm uses soft decision decoding to greatly aid the correct detection of a block of received symbols.

At the transmitter, the LDPC encoder combines with the DSQ128 symbol mapper to set the LDPC block (PHY frame) of data sent. The LDPC decoder uses soft decisions to recover the LDPC block data sent. DSQ128 symbols are demapped into 7-bit data. A CRC-8 is computed across the recovered data bits in each PHY frame and compared to the transmitted value to detect block errors.



3.2.2.1.13 Four-Channel PAM16 Soft LDP Decision and LDPC Decoder (IEEE Std. 802.3bz)

The objective of the matrix equalization, matrix crosstalk cancellation, and the clock recovery processes is to reconstruct, as closely as possible similar to 10GBASE-T, where the original PAM16 symbol is transmitted by the link partner. The LDPC decoding algorithm uses soft decision decoding to greatly aid the correct detection of a block of received symbols.

At the transmitter, the LDPC encoder combines with the PAM16 symbol mapper to set the LDPC block (PHY frame) of data sent. The LDPC decoder uses soft decisions to recover the LDPC block data sent. PAM16 symbols are demapped into 4-bit data.

3.2.2.1.14 Deframe and Descramble

After LDPC decoding, PHY-level framing is no longer required as PHY-level frames are not visible to higher layers. All non-payload bits are removed by the deframing operation prior to transmission to the MAC. Each recovered PHY-level frame contains sufficient payload data to create 50 consecutive 65-bit XGMII data/control blocks. The recovered payload data bits are descrambled with the same 58-bit scrambler polynomial used at the transmitter. The device uses a side-stream scrambler that is synchronized by the Physical Coding Sublayer (PCS) receive block.

3.2.3 100BASE-TX and 1000BASE-T Modes

3.2.3.1 Encoder

In 100BASE-TX mode, the PHY transmits a continuous data stream over the twisted-pair cable. The transmit packet is encapsulated by replacing the first two nibbles of preamble with a start-of-stream delimiter

(J/K codes) and appending an end-of-stream delimiter (T/R codes) to the end of the packet. The transmitter repeatedly sends the idle code groups between packets. The encoded data stream is serialized and then scrambled by the stream cipher block. The scrambled data is then encoded into MLT3 signal levels.

In 1000BASE-T mode, the PHY simultaneously transmits and receives a continuous data stream on all four pairs of the Category 5 cable. Byte-wide data from the transmit data pins is scrambled when the transmit enable is asserted and the trellis code is modulated into a four-dimensional constellation, each dimension of which is transmitted on a separate channel of the transmit data stream. The transmit packet is encapsulated by replacing the first two bytes of the preamble with a start-of-stream delimiter and appending an end-of-stream delimiter to the end of the packet. When the transmit error input is asserted during a packet transmission, a transmit error code group is sent in place of the corresponding data code group. The transmitter sends idle code groups or carrier extend code groups between packets. Carrier extension is used by the MAC to separate packets within a multiple-packet burst, and is indicated by asserting the transmit error signal and placing 0x0F on the transmit data pins while the transmit enable is low. A carrier extend error is indicated by replacing the transmit data input with 0x1F during carrier extension.

The transmitter encoding for each of the data rates complies with the applicable IEEE standard.



3.2.3.2 Decoder

In 100BASE-TX mode, following equalization and clock recovery, the receive data stream is converted from MLT3 to serial non-return to zero (NRZ) data. The NRZ data is descrambled by the stream cipher block. The descrambled data is then deserialized and aligned into 5-bit code groups. The 5-bit code groups are decoded into 4-bit data nibbles. The start-of-stream delimiter is replaced with preamble nibbles, and the end-of-stream delimiter and idle codes are replaced with 0x0. When an invalid code group is detected in the data stream, the PHY flags the error with an in-band status code that indicates error.

In the 1000BASE-T mode, following clock recovery, equalization, crosstalk and echo cancellation and gain adjustment, the data stream is passed through the Viterbi decoder, descramble, and translated back into byte-wide data

The start-of-stream delimiter is replaced with preamble bytes, and the end-of-stream delimiter and idle codes are replaced with 0x00. Carrier extend codes are replaced with 0x0F or 0x1F. Decoding complies with IEEE 802.3ab standard.

3.2.3.3 Carrier Sense

Carrier sense is reported with the in-band status mechanized according to the standard. In 1000BASE-T SGMII mode, the carrier sense information is encoded into the control signals.

3.2.3.4 Link Monitor

In 100BASE-TX mode, receive signal energy is detected by monitoring the receive pair for transitions in the signal level. Signal levels are qualified using squelch detect circuits. When no signal is detected on the receive pair, the link monitor enters the link fail state and the transmission and reception of data packets are disabled. When a valid signal is detected on the receive pair for a minimum of 1 ms, the link monitor enters link-pass state and the transmit and receive functions are enabled.

Following the auto-negotiation in the 1000BASE-T mode, the master transceiver begins sending data on the media. The slave transceiver also begins transmitting when it has recovered the master transceiver's timing. To complete the process, each end of the link continuously monitors its local receiver status. The master begins training its receiver when it detects the slave's transmitted signal. When the local receiver status has been good for at least 1 μ s, the link monitor enters the link-pass state and the transmission and reception of data packets are enabled. When the local receiver status is bad for more than 750 ms, the link monitor enters the link-fail state, and the transmission and reception of data packets are then disabled.



3.2.4 Digital Adaptive Equalizer (100/1000BASE-T Only)

The digital adaptive equalizer removes ISI created by the transmission channel media. The equalizer accepts sampled unequalized data from the ADC on each channel and produces equalized data. The PHY achieves optimum signal-to-noise ratio by using a combination of Feed-Forward Equalization (FFE) and Decision-Feedback Equalization (DFE) techniques. Under harsh noise environments, these powerful techniques achieve a Bit Error Rate (BER) of less than 1×10^{-12} for transmissions up to 100 m on Category 5 twisted-pair cabling. The all-digital nature of the design makes the performance very robust. The filter coefficients are self-adapting to accommodate varying conditions of cable quality and cable length.

3.2.4.1 Echo Canceler

Because of simultaneous bidirectional transmission of data on each cable pair in 1000BASE-T, echo impairments from the local transmitters are present at each receiver. A digital echo canceler is used to subtract an estimate of the echo from the received signal at the output of the FFE. The digital echo canceler coefficients are adaptive to track the varying echo transfer function caused by different channels, transmitters, cable geometry, and environmental conditions.

3.2.4.2 Crosstalk Canceler

The PHY transmits and receives a continuous data stream on four channels. For a given channel, the signals sent by the other three local transmitters cause impairments on the received signal because of near-end crosstalk (Next) between the pairs. The effect can be canceled because each receiver has access to the data for the other three pairs that cause this interference. The output of the adaptive Next canceling filters is added to the FFE output to cancel the NEXT impairment.

3.2.4.3 Analog-to-Digital Converter

Each receive channel has its own 125 MHz ADC that samples the incoming data on the receive channel and feeds the output to the digital adaptive equalizer. Advanced analog circuit techniques achieve the following results:

- Low offset
- High-power supply noise rejection
- Fast settling time
- Low bit error rate

3.2.4.4 Clock Recovery/Generator

The clock recovery and generator block creates the transmit and receive clocks for 1000BASE-T and 100BASE-TX operation. In 100BASE-TX mode, the transmit clock is locked to a multiple of the 50 MHz crystal input, and the receive clock is locked to the incoming data stream.



In 1000BASE-T mode, the two ends of the link perform loop timing. One end of the link is configured as the master, and the other is configured as the slave. The master transmit and receive clocks are locked to a multiple of the 50 MHz crystal input. The slave transmit and receive clocks are locked to the incoming receive data stream. Loop timing allows for the straightforward cancellation of echo and Next impairments by ensuring that the transmitter and receiver at each end of the link are operating at the same frequency.

3.2.4.5 Baseline Wander Correction

The 1000BASE-T and 100BASE-TX data streams are not always DC-balanced and thus have spectral energy down to DC; however, magnetics introduce high-pass filtering that removes some of this low-frequency energy. The loss of this low-frequency energy causes a form of inter-symbol interference called baseline wander. Baseline wander is an apparent time-varying and pattern-dependent shift of the DC offset of the signal that is actually due to the absence of DC/low-frequency energy caused by the high-pass filtering in the system. The presence of this type of inter-symbol interference can greatly reduce the noise immunity and performance of the receiver. The PHY automatically compensates for baseline wander by restoring the missing low-frequency energy to the receive signal, which significantly improves the performance of the receiver and greatly reduces the probability of symbol errors.

3.2.4.6 Multimode Tx Digital-to-Analog Converter

The multimode transmit DAC transmits PAM5- and MLT3-coded symbols. The transmit DAC performs signal wave shaping that decreases the unwanted high-frequency signal components and reduces electromagnetic interference (EMI). The transmit DAC uses a current drive output that is well-balanced, to produce low-noise transmit signals.

3.2.4.7 Stream Cipher

In 1000BASE-T and 100BASE-TX modes, the transmit data stream is scrambled to reduce radiated emissions and to ensure that there are adequate transitions within the data stream. The 1000BASE-T scrambler also ensures that there is no correlation among symbols on the four different wire pairs and in the transmit and receive data streams. The scrambler reduces peak emissions by spreading the signal energy over the transmit frequency range and eliminating peaks at certain frequencies. The randomization of the data stream also assists the digital adaptive equalizers and echo/crosstalk cancelers. The algorithms in these circuits require there to be no sequential or cross-channel correlation among symbols in the various data streams.

In 100BASE-TX mode, the transmit data stream is scrambled by exclusive-ORing the encoded serial data stream. This is done with the output of an 11-bit-wide Linear Feedback Shift Register (LFSR), producing a 2047-bit non-repeating sequence.

In 1000BASE-T mode, the transmit data stream is scrambled by exclusive-ORing the input data byte with an 8-bit-wide cipher text word. The cipher text word generates each symbol period from eight uncorrelated maximal length data sequences that are produced by linear remappings of the 33-bit-wide LFSR output. After the scrambled data bytes are encoded, the sign of each transmitted symbol is again randomized by a 4-bit-wide cipher text word that is generated in the same manner as the 8-bit word. The master and slave transmitters use different scrambler sequences to generate the cipher text words. For repeater or switch applications, where all ports can transmit the same data simultaneously, signal energy is randomized further by using a unique seed to initialize the scrambler sequence for each PHY.



The receiver descrambles the incoming data stream by exclusive-ORing it with the same sequence generated at the transmitter. The descrambler detects the state of the transmit LFSR by looking for a sequence representing consecutive idle code groups. The descrambler locks to the scrambler state after detecting a sufficient number of consecutive idle codes. The PHY enables transmission and reception of packet data only when the descrambler is locked. The receiver continually monitors the input data stream to ensure that it has not lost synchronization by checking that inter-packet gaps containing idles or frame extensions are received at expected intervals and by monitoring idle error rate during idles. When the PHY detects loss of synchronization, it notifies the remote PHY of the inability to receive packets (1000BASE-T mode only) and attempts to resynchronize to the received data stream. If the descrambler is unable to resynchronize for a period of 750 ms, the PHY is forced into the link-fail state.

3.2.4.8 Wire Map and Pair Skew Correction

During 1000BASE-T operation, the PHY has the ability to automatically detect and correct some UTP cable wiring errors. The symbol decoder detects and compensates for (internal to the PHY) the following errors:

- Wiring errors caused by the swapping of pairs within the UTP cable.
- Polarity errors caused by the swapping of wires within a pair.

The PHY also automatically compensates for differences in the arrival times of symbols on the four pairs of the UTP cable. The varying arrival times are caused by differing propagation delays (commonly referred to as delay skew) between the wire pairs. The PHY can tolerate delay skews of up to 64 ns. Auto-negotiation must be enabled to take advantage of the wire map correction. During 100 Mb/s operation, the pair swaps are corrected. Note that the delay skew is not an issue because only one pair of wires is used in each direction.

3.2.4.9 Automatic MDI Crossover

Note: This function only operates when copper auto-negotiation is enabled.

During copper auto-negotiation, one end of the link must perform an MDI crossover so that each transceiver's transmitter is connected to the other receiver. The PHY can perform an automatic MDI crossover, eliminating the need for crossover cables or cross-wired (MDIX) ports. During auto-negotiation, the PHY normally transmits on the TRD±{0} pin and receives on the TRD±{1} pin.

When connecting to another device that does not perform MDI crossover, the PHY automatically switches its TRD± pin pairs, when necessary, to communicate with the remote device. When connecting to another device that does have MDI crossover capability, an algorithm determines which end performs the crossover function.

During 1000BASE-T operation, the PHY swaps the transmit symbols on pairs 0 and 1, and pairs 2 and 3 if auto-negotiation completes in the MDI crossover state. The 1000BASE-T receiver automatically detects pair swaps on the receive inputs and aligns the symbols properly within the decoder. During 100BASE-TX and 1000BASE-T operation, pair swaps automatically occur within the device and do not require user intervention.

3.2.4.10 100BASE-TX Forced Mode Auto-MDIX

Note: This function only operates when copper auto-negotiation is disabled.



This feature enables users to disable copper auto-negotiation in 100BASE-TX and still take advantage of the automatic MDI crossover function. Each time the forced link is down for a least four seconds, auto-negotiation is internally enabled with its automatic MDI crossover function active until link pulses or 100TX idles are detected. Once detected, the PHY returns to forced mode operation.

Users should set the same speed in register 0 and the auto-negotiation advertisement register 4. For example, a user wanting to force 100TX full-duplex should write register 0 with 0x2100 and register 4 with 0x0181. The feature is enabled by writing register 0x18, shadow 7, bit 9 = 1. Copper link can be determined by reading register 0x19, bit 2.

3.2.5 Auto-negotiation

3.2.5.1 Copper Interface

The PHY negotiates its mode of operation over the copper media using the auto-negotiation mechanism, defined in the IEEE Std. 802.3u, IEEE Std. 802.3ab, and IEEE Std. 802.3an specifications. When auto-negotiation is enabled, the PHY automatically chooses the mode of operation by advertising its abilities and comparing them with those received from its link partner. The PHY can be configured to advertise the following modes:

- 10GBASE-T full-duplex
- 5GBASE-T full-duplex
- 2.5GBASE-T full-duplex
- 1000BASE-T full-duplex
- 100BASE-TX full-duplex

The transceiver negotiates with its link partner and chooses the highest common operating speed and duplex mode commonly referred to as Highest Common Denominator (HCD). Auto-negotiation can be enabled or disabled by hardware and software control, but is always required for 1000BASE-T and 10GBASE-T operation.

3.2.6 Energy Efficient Ethernet (EEE)

The PHY features the EEE protocol for 10GBASE-T, 5GBASE-T, 2.5BASE-T and 1000BASE-T speeds. EEE enables the PHY to auto-negotiate and operate with EEE-compliant link partners to reduce overall system power during low use of the link. The PHY is compliant with IEEE Std. 802.3az-2010. The EEE function is enabled by advertising this capability in the EEE Advertisement Register (DEVAD = 7, Register 0x003C) and then, if the link partner also supports EEE, by appropriately signaling with a transition from Idle to Low-power Idle over XFI (10GBASE-KR) interfaces as defined and described in the draft specification.

The PHY offers a basic native EEE mode of operation for switches and MACs that support LPI signaling across the XFI (10GBASE-KR) or SGMII interface.

In native mode, the PHY supports the following:

Note: Auto-negotiation must be enabled for all interfaces.

- 1000BASE-T
- 2.5GBASE-T



- 5GBASE-T
- 10GBASE-T

This low power mode allows for systems on both sides of the link to disable portions of their functionality and save power during periods of low link use.

EEE also specifies the means for link partners to determine whether EEE is supported and to provision the best set of parameters common to both devices.

3.2.6.1 IEEE EEE Mode

The low-power operational mode is an optional mode that enables power saving by switching off parts of the communication device's functionality when no data is transmitted or/and received.

The decision on whether the system should enter Low Power Idle (LPI) mode or exit LPI mode is done at the MAC level and is communicated to the PHY level to enable maximum power savings. [Figure 3-5](#) shows a conceptual sketch of the decision flow and agents involved.

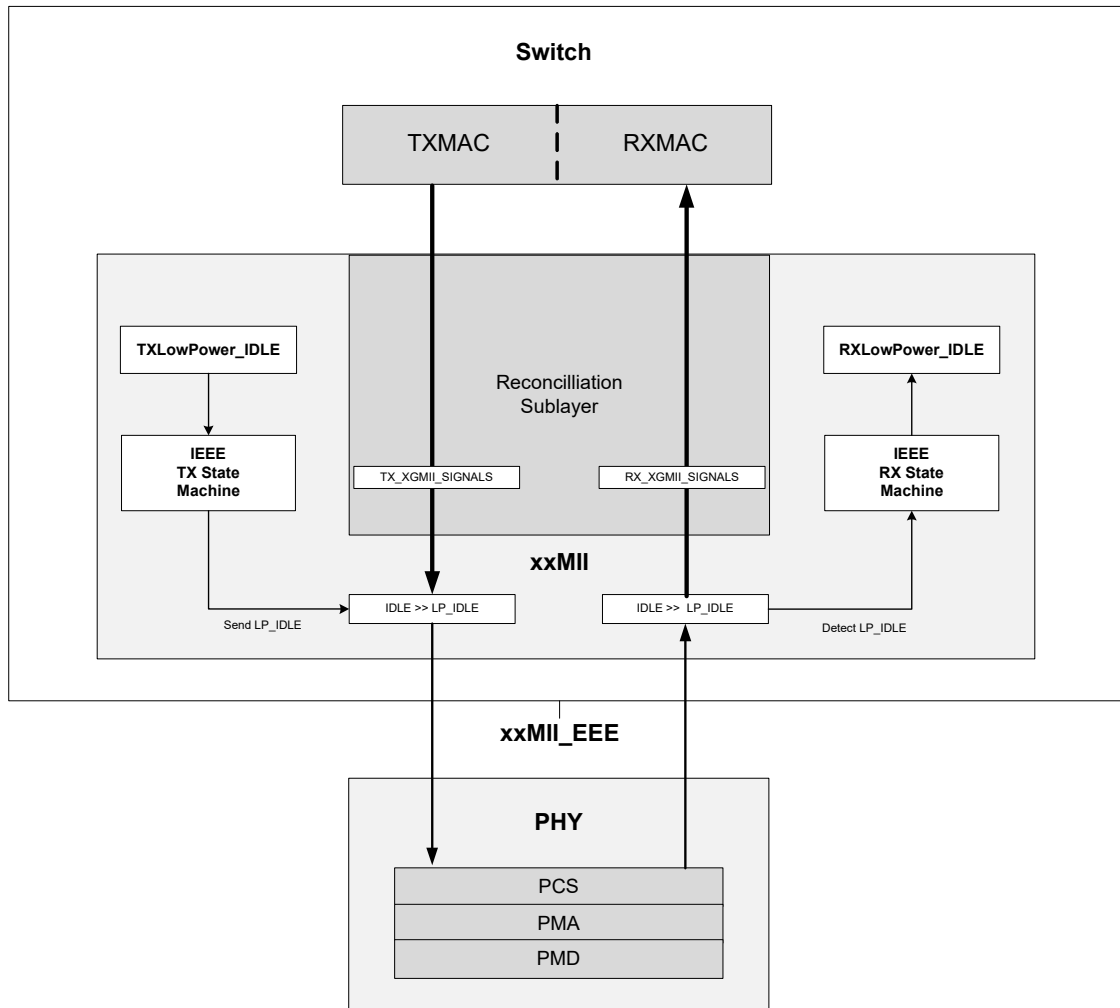


Figure 3-5. IEEE Energy Efficient Ethernet Mode: MAC/PHY Functional Blocks



3.2.6.1.1 Transmit Direction

In the transmit direction, the entrance to LPI mode of operation is triggered by the reception of LP_IDLE code words on the MAC interface. Following the reception of the LP_IDLE code word, PHY transmits special Sleep signal to communicate to the link partner that the local system is entering LPI mode.

After the Sleep signal transmission, the transmit function of the local PHY enters a quiet mode.

Periodically, the transmit function of the local PHY is enabled to transmit refresh signals that are used by the link partner to update adaptive filters and timing circuits in order to maintain link integrity. This quiet-refresh cycle continues until local MAC sends IDLE code words, which signals to the PHY that low power idle mode should end.

The transmit function in the PHY communicates this to the link partner and then resumes normal operation mode after a predefined period of time.

In LPI mode, the PHY device is not immediately available for data transmission requests. The PHY awakens by sending normal idle codes on the MAC interface. After receiving the Idle code on the MAC interface, the PHY starts the waking-up process. The maximum PHY recovery time (T_w) is different for different protocols. For 10GBASE-T, T_w is 4.48 μs . Note that in 10GBASE-T, WAKE time (T_w) includes the alert time. Figure 3-6 shows the time diagram of the transmission from a transmitter perspective.

When the link is established in 100BASE-TX and the MAC is transmitting idles, the PHY starts sending LPI. When the link is established in 1000BASE-T, the PHY enters LPI mode only after it transmits LP_SLEEP and receives LP_SLEEP from the link partner. The PHY keeps the link related parameters up to date during refresh. The MAC determines when the PHY stops sending LPI signals on the MDI and transition to normal mode. The PHY stops transmitting LPI signals when the PHY receives normal idles from the MAC. After the PHY stops transmitting LPI signals and after $T_w = 20.5 \mu s$ for 100BASE-TX or $T_w = 16.5 \mu s$ for 1000BASE-T link, the MAC might send normal data.

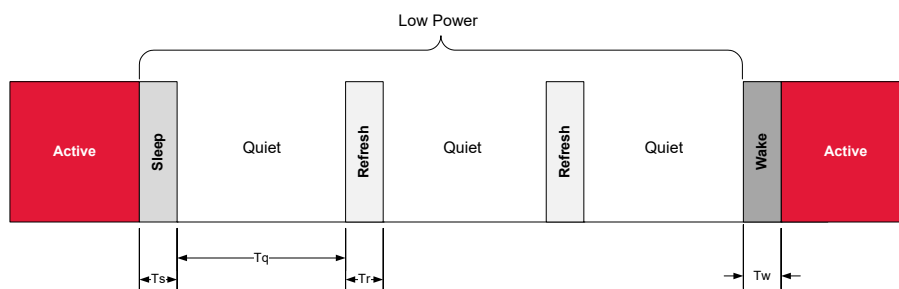


Figure 3-6. LPI Mode: Transmit Signal Diagram

3.2.6.1.2 Receive Direction

In the receive direction, the LPI operation is triggered by the reception of LP_IDLE code words from the PHY link partner. This signals that the link partner is about to enter LPI mode.

After sending the LP_IDLE code word, the link partner ceases transmission and enters quiet mode. During this time, the local receiver can disable some of the functionality to reduce power consumption.



Periodically, the link partner transmits refresh signals that are used by the local PHY to update adaptive coefficients and timing circuits.

This quiet-refresh cycle continues until the link partner initiates a transition back to full data mode by transmitting (in 10GBASE-T, an ALERT signal followed by) a WAKE signal that enables the local receiver to prepare for the full data rate. After a system-specified recovery time, the link supports a nominal operational data rate.

In 100BASE-TX mode, the local PHY transmits a special LP_SLEEP signal to communicate to the link partner that the local system is entering LPI mode. In 1000BASE-T mode, the transmit function of the local PHY enters a quiet mode only after the local PHY transmits LP_SLEEP and receives LP_LP_SLEEP from the remote PHY. If the remote PHY does not signal LPI, neither PHY can go quiet. The LPI requests are still passed from one end to the other end of the link, since other system energy saving may be achieved even if the PHY link does not go quiet.

The T_w waiting default for 100BASE-TX is 20.5 μ s and for 1000BASE-T is 16.5 μ s, unless through Link Layer Data Protocol (LLDP) the upper layer MACs have negotiated new values.

No data frames are lost or corrupted during the transition to or from EEE mode.



3.2.7 Physical Layer (PHY)

The X710-TM4 provides up to four Ethernet MAC ports with integrated PHY interfaces to connect either directly to the medium (backplane or direct attached twinaxial copper cable assemblies) or to external PHYs. The X710-TM4 Ethernet physical interfaces are multi-rate Medium Attachment Unit Interfaces (MAUI) that can be configured for operation at 10 Gb/s, 5 Gb/s, 2.5 Gb/s, 1 Gb/s or 100 Mb/s link speeds. The X710-TM4 supports eight physical high speed SerDes lanes, each capable of operating at up to 10.3125 GBaud. Four SerDes lanes are connected to the internal PHY and are not externally accessible. The other four SerDes lanes are for external connections (connect up to two SFP+ modules or to an external dual-port PHY). The X710-TM4 also supports auto-negotiation when configured for backplane Ethernet to automatically select between SGMII, SFI, 1000BASE-KX and 10GBASE-KR. The X710-TM4 supports polarity correction on the MAUI interfaces to ease board routing when using integrated PHYs.

3.2.7.1 MAC and MAUI Port Configuration

The X710-TM4 supports up to four Ethernet MAC ports and the corresponding MAUI / physical interfaces can be configured to different speeds and protocols as shown in [Figure 3-7](#).

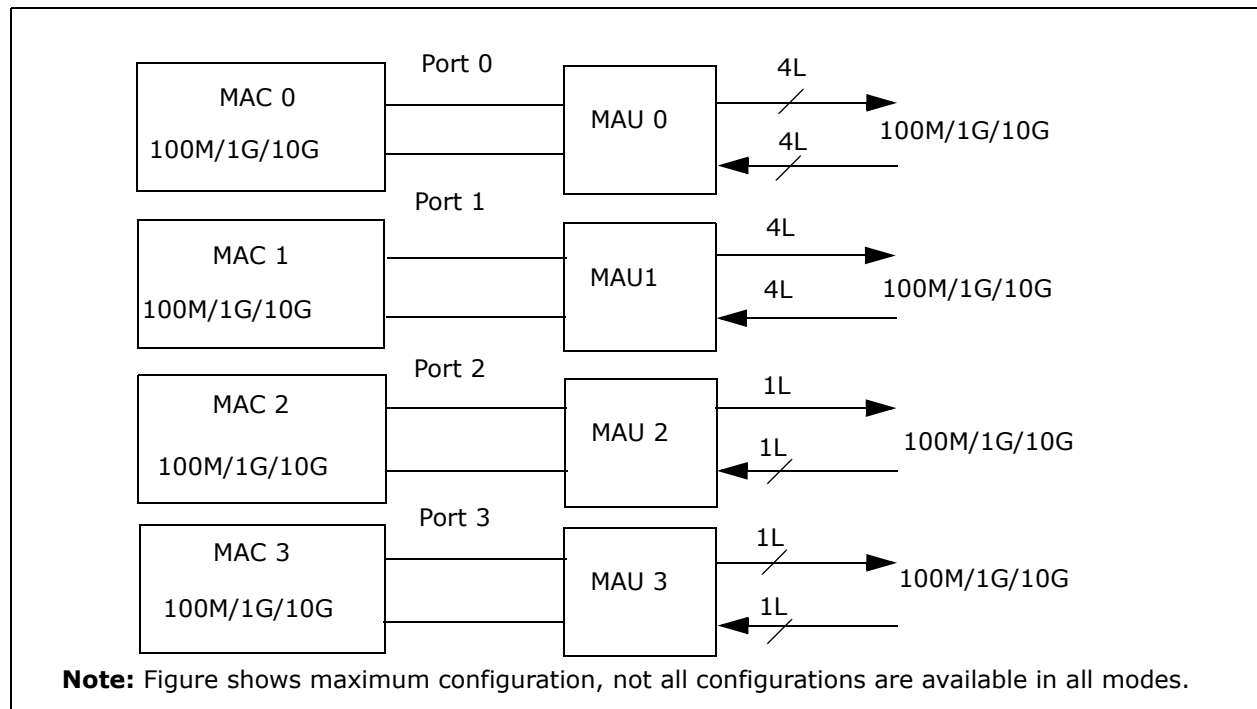


Figure 3-7. X710-TM4 MAC and MAUI Interfaces

When in 10 Gb/s operating mode, the MAUI interface for each of the Ethernet ports 0 through 3 can be independently configured for one of the following physical interfaces:

- A single lane 10GBASE-KR interface on up to four Ethernet ports for direct connectivity to backplane medium.



- A single lane SFI (XFI+) interface connectivity to 10GBASE-LR/SR optical modules or for direct connectivity for up to 7 m twin-axial copper cable assemblies.

When in 1 Gb/s operating mode, the MAUI interface for each of the Ethernet ports 0 through 3 can be independently configured for one of the following physical interfaces:

- A single lane 1000BASE-KX interface for direct connectivity to backplane medium.
- A single lane SGMII (1 Gb/s) over a KX compliant electrical interface.

When in 100 Mb/s operating mode, the MAUI interface for each of the Ethernet ports 0 through 3 can be independently configured for one of the following physical interfaces:

- A single lane SGMII (100 Mb/s) over a KX compliant electrical interface.

Table 3-27. Supported Electrical Modes

Speed	Mode	PMD Specification
10G	10GBASE-KR	IEEE Std. 802.3-2018 Clause 72
	10GBASE-T	802.3ae
	SFI	SFF 8431
1G	1000BASE-KX	IEEE Std. 802.3-2018 Clause 36
	SGMII	IEEE Std. 802.3-2018 Clause 36
100M	SGMII	IEEE Std. 802.3-2018 Clause 36

When any of the Ethernet port is directly connected to the medium (backplane or twin-axial copper cable assembly) auto-negotiation protocol runs on MAUI lane 0 only, as per the IEEE Clause 73 requirements.

In dual port (four lane per port) configuration (10G/1G: KR/KX) AN needs to run on lane 0 of the 4 lanes per port.

In quad port (one lane per port) configuration (10G/1G: KR/KX) AN needs to run on all lanes.

Table 3-28 lists the possible speed, interface type and lane configurations supported by the X710-TM4 on Ethernet MAC ports 0 through 3.

Table 3-28. Port, Speed, Interface Type, and Lane Configuration

Port Num	Speed Config	Num of Lanes in Each Direction	Port Type Config	Rate Per Lane (GBaud)	MAU Lanes Used
Port 0	10 Gb/s 1 Gb/s 100 Mb/s	1	KR, SFI, KX, SGMII	10.3125, 1.25	Lane0
Port 1	10 Gb/s 1 Gb/s 100 Mb/s	1	KR, SFI, KX, SGMII	10.3125, 1.25	Lane0
Port 2	10 Gb/s 1 Gb/s 100 Mb/s	1	KR, SFI, KX, SGMII	10.3125, 1.25	Lane0
Port 3	10 Gb/s 1 Gb/s 100 Mb/s	1	KR, SFI, KX, SGMII	10.3125, 1.25	Lane0



3.2.7.1.1 MAU Logical Lanes to Physical Pin Configuration

3.2.7.1.2 Logical Lanes-to-Physical Pin Configuration

The X710-TM4 provides the following combinations of logical-to-physical lane configurations to enable flexible board routing for blade systems and LOMs (Figure 3-8). See Section 3.2.7.2 for logical lane-to-physical pin mapping for the supported port configurations.

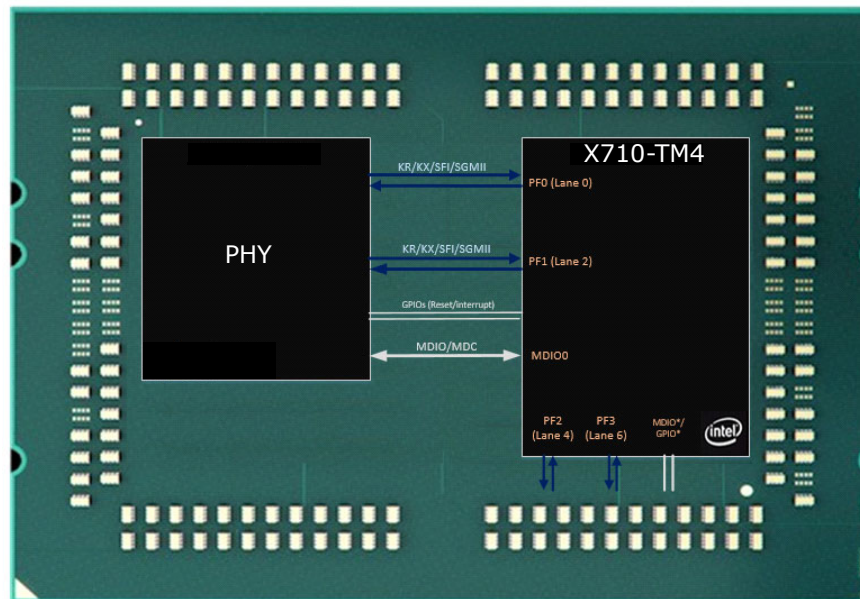


Figure 3-8. Logical Lane-to-Physical Lane Mapping

3.2.7.2 Possible Port-to-Physical Lane Configurations for Various PHY Interfaces

See Table 3-31 for possible port to physical lane configurations allowed by the X710-TM4 for various PHY interfaces for single, dual and quad port configurations.

3.2.7.3 10 Gb/s Interfaces

The X710-TM4 provides complete functionality to support up to four 10 Gb/s ports. The device performs all functions required for transmission and reception defined in the various standards.

A PHY interface is included to attach either to an external PMA or Physical Medium Dependent (PMD) components.

The X710-TM4 enables 10 Gb/s operation compliant to the KR or SFI specifications.



3.2.7.3.1 10GBASE-KR Operating Mode

The KR interface supports data rates of 10 Gb/s over copper traces in improved FR4 PCBs. Data is transferred over a single differential path in each direction for a total of two pairs, with each path operating at 10.3125 Gbaud \pm 100 ppm to support overhead of 64B/66B coding. The interface is used to connect the X710-TM4 to a KR switch port over the backplane or to an external PHY.

The MAUI interface is configured as a KR interface while auto-negotiation to a KR link partner is detected. KR operation can also be forced by firmware (see [Section 3.2.9](#)). When in 10GBASE-KR operating mode, MAUI lane 0 is used for receive and transmit activity while lanes 1 to 3 of the MAUI interface are powered down.

When configured to operate in 10GBASE-KR mode the X710-TM4 supports IEEE Std. 802.3az EEE operation. See [Section 5.3.1](#) for more detail.

3.2.7.3.1.1 KR Overview

10GBASE-KR definition enables 10 Gb/s operation over a single differential path in each direction for a total of two pairs, or four connections. This system uses the 10GBASE-KR PCS as defined in IEEE Std. 802.3-2018 Clause 49 with amendments for auto-negotiation specified in IEEE Std 802.3ap and 10 Gigabit PMA as defined in IEEE Std 802.3 Clause 51. The 10GBASE-KR PMD is defined in IEEE Std. 802.3ap Clause 72. The 10GBASE-KR PHY includes 10GBASE-KR FEC, as defined in IEEE Std 802.3ap Clause 74. FEC support is optional and is negotiated between Link partners during auto-negotiation as defined in IEEE Std 802.3ap Clause 73. Activating FEC improves link quality (2dB coding gain) by enabling correction of up to 11 bit-burst errors.

KR is a full-duplex interface that uses a single self-clocked serial differential link in each direction to achieve 10 Gb/s data throughput. The serial link transfers scrambled data at 10.3125 Gbaud to accommodate both data and the overhead associated with 64B/66B coding. The self-clocked nature eliminates skew concerns between clock and data, and enables a functional reach of up to one meter.

Following initialization and auto-negotiation 10GBASE-KR defines a start-up protocol, where link partners exchange continuous fixed length training frames using differential Manchester Encoding (DME) at a signaling rate equal to one quarter of the 10GBASE-KR signaling rate. This protocol facilitates timing recovery and receive equalization while also providing a mechanism through which the receiver can tune the transmit equalizer to optimize performance over the backplane interconnect. Successful completion of the start-up protocol enables transmission of data between the link partners.

[Figure 3-9](#) shows the architectural positioning of 10GBASE-KR.

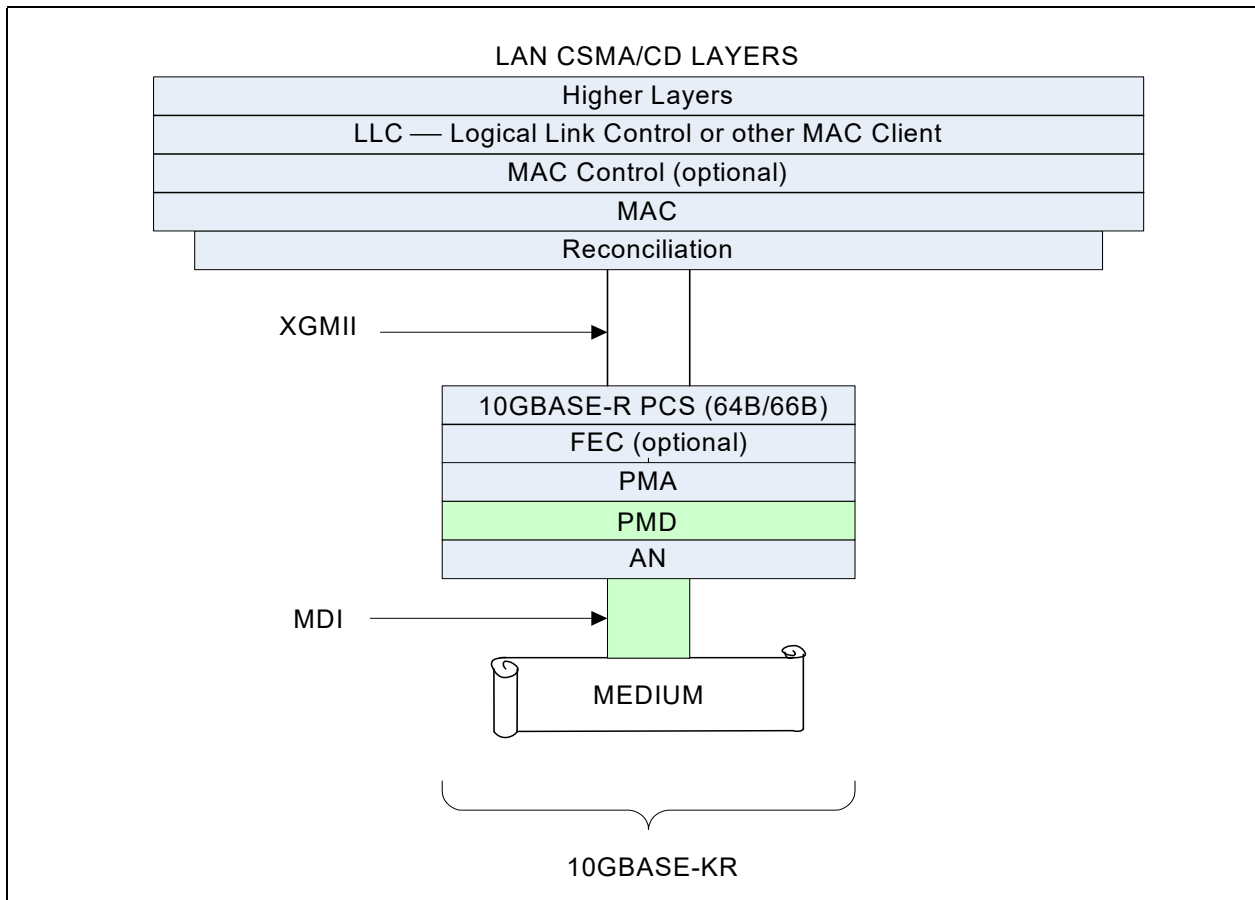
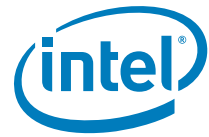


Figure 3-9. Architectural Positioning of 10GBASE-KR

3.2.7.3.1.2 KR Electrical Characteristics

The KR lane is a low swing AC coupled differential interface using NRZ signaling. AC coupling allows for interoperability between components operating from at different supply voltages. Low swing differential signaling provides noise immunity and improved reduced EMI. Differential signal swings defined specifications depend on several factors, such as transmitter pre-equalization and transmission line losses.

The KR signal paths are point-to-point connections. Each path corresponds to a KR lane and is comprised of two complementary signals making a balanced differential pair. There is a single differential path in each direction for a total of two pairs, or four connections.

The 10GBASE-KR link requires a nominal 100 Ω differential source and load terminations with AC coupling on the receive side. The signal paths are intended to operate up to approximately one meter, including two connectors, over controlled impedance traces on improved FR4 PCBs.



3.2.7.3.1.3 KR Reverse Polarity

The X710-TM4 supports reverse polarity of the KR transmit and receive lanes. It is enabled by NVM settings in the Core 0/1 Analog Configuration Modules.

3.2.7.3.2 SFI Operating Mode

The MAUI interface is configured as SFI by firmware (see [Section 3.2.13](#)). When in SFI operating mode, only the operation of the X710-TM4 Analog Front End (AFE) is modified, while the rest of the X710-TM4 logic and circuitry operates similar to 10GBASE-KR.

When configured to operate in SFI mode, the X710-TM4 supports IEEE Std. 802.3az EEE operation (see [Section 5.3.1.7.3](#) for further information).

3.2.7.3.2.1 SFI Overview

SFI definition enables 10 Gb/s operation over a single differential path in each direction for a total of two pairs, or four connections. When in SFI operating mode the X710-TM4 uses the 10GBASE-R PCS and 10 Gigabit PMA as defined in IEEE Std. 802.3-2018 Clause 49 and 51, respectively.

SFI is a full-duplex interface that uses a single self-clocked serial differential link in each direction to achieve 10 Gb/s data throughput. The serial link transfers scrambled data at 10.3125 GBaud to accommodate both data and the overhead associated with 64B/66B coding. The self-clocked nature eliminates skew concerns between clock and data.

3.2.7.3.2.2 SFI Electrical Characteristics

The SFI lane is a low swing AC coupled differential interface using NRZ signaling. AC coupling allows for interoperability between components operating from at different supply voltages. Low swing differential signaling provides noise immunity and improved reduced EMI. Differential signal swings defined specifications depend on several factors, such as transmitter pre-equalization and transmission line losses.

The SFI signal paths are point-to-point connections. Each path corresponds to a SFI lane and is comprised of two complementary signals making a balanced differential pair. There is a single differential path in each direction for a total of two pairs, or four connections. The signal paths are intended to operate on FR4 PCBs.

SFI interface typically operates over 200 mm of improved FR4 material or up to about 150 mm of standard FR4 with one connector. The electrical interface is based on high speed low voltage AC coupled logic with a nominal differential impedance of 100 Ω . The SFI link requires nominal 100 Ω differential source and load terminations on both the host board and the module. The SFI terminations provide both differential and common mode termination to effectively absorb differential and common mode noise and reflections. All SFI transmitters and receivers are AC coupled. SFP+ modules incorporate blocking capacitors on all SFI lines.



3.2.7.4 1 Gb/s Interface

The X710-TM4 provides complete support for up to two 1 Gb/s port implementations. The device performs all functions required for transmission and reception defined by the different standards.

A lower-layer PHY interface is included to attach either to external PMA or PMD components.

The X710-TM4 enables 1 Gb/s operation compliant with the KX or SGMII specifications.

3.2.7.4.1 1000BASE-KX Operating Mode

The MAUI interface, when operating as a KX interface, supports data rates of 1 Gb/s over copper traces on improved FR4 PCBs. Data is transferred over a single differential path in each direction for a total of two pairs (Lane 0 of MAUI interface and Lanes 1 to 3 powered down), with each path operating at 1.25 GBaud to support overhead of 8B/10B coding. The interface is used to connect the X710-TM4 to a KX compliant switch port over the backplane or to KX compliant 1 Gb/s PHY device. In the event of auto-negotiation defined in IEEE Std. 802.3ap clause 73 ending with 1 Gb/s as the HCD, the MAUI interface is configured as a KX interface. KX operating mode can also be forced by firmware (see [Section 3.2.13](#)).

3.2.7.4.1.1 KX Overview

1000BASE-KX extends the family of 1000BASE-X PHY signaling systems. KX specifies operation at 1 Gb/s over two differential, controlled impedance pairs of traces (one pair for transmit, one pair for receive). This system uses the 1000BASE-X PCS and PMA as defined in IEEE Std. 802.3-2018 Clause 36 together with the amendments placed in IEEE Std. 802.3ap. The 1000BASE-KX PMD is defined in IEEE Std. 802.3ap Clause 70.

KX is a full-duplex interface that uses a single serial differential link in each direction to achieve 1 Gb/s data throughput. Each serial link operates at 1.25 GBaud to accommodate both data and the overhead associated with 8B/10B coding. The self-clocked nature eliminates skew concerns between clock and data, and enables a functional reach of up to one meter. The X710-TM4 also supports IEEE Std. 802.3az EEE operation when configured to operate in 1000BASE-KX mode (see [Section 3.2.7.4.1](#) for further information).

Figure 3-10 shows the architecture positioning of 1000BASE-KX.

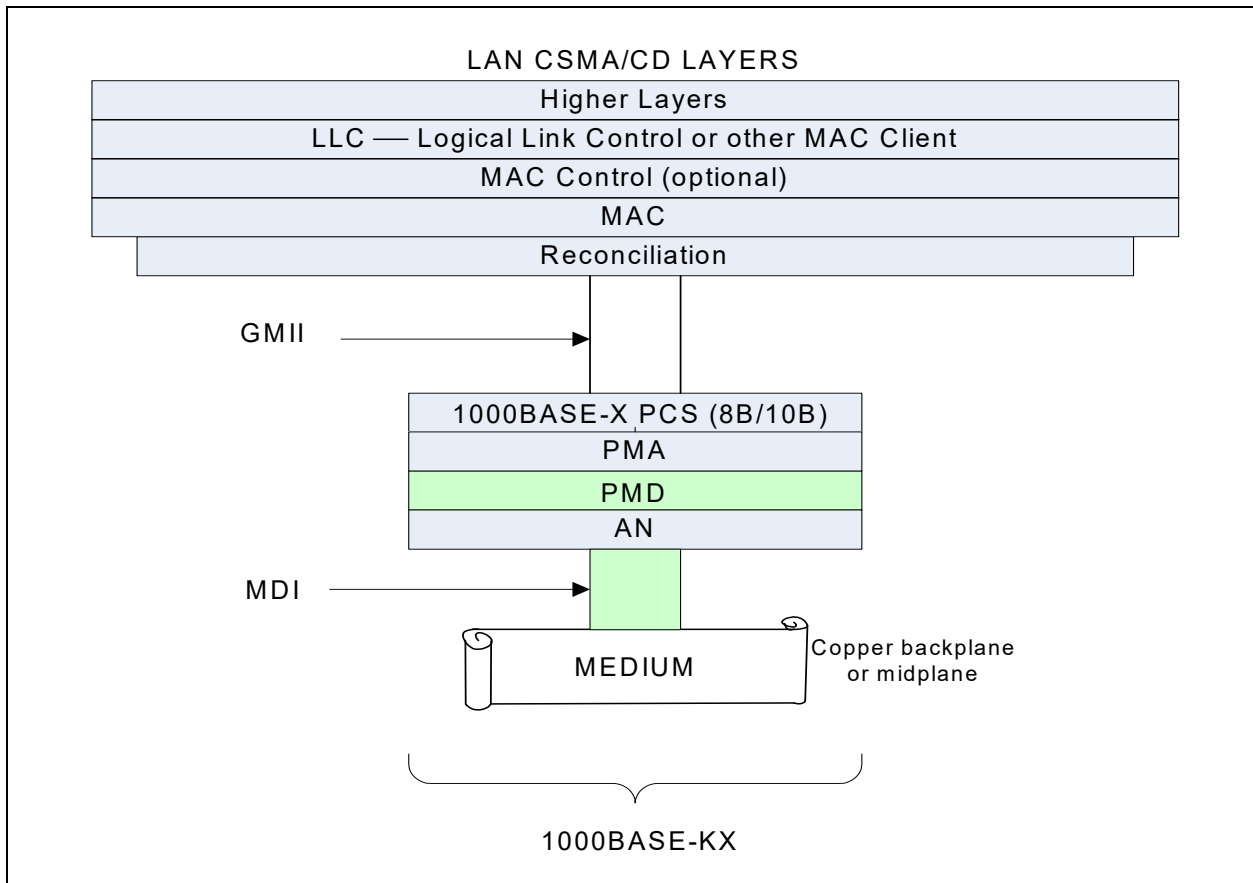


Figure 3-10. Architectural Positioning of 1000BASE-KX

3.2.7.4.1.2 KX Electrical Characteristics

The KX lane is a low swing AC coupled differential interface using NRZ signaling. AC coupling allows for interoperability between components operating from at different supply voltages. Low swing differential signaling provides noise immunity and improved reduced EMI. Differential signal swings defined specifications depend on several factors, such as transmitter pre-equalization and transmission line losses.

The KX signal paths are point-to-point connections. Each path corresponds to a KX lane and is comprised of two complementary signals making a balanced differential pair. There is one differential path in each direction for a total of two pairs, or four connections. The signal paths are intended to operate up to approximately one meter over controlled impedance traces on improved FR4 PCBs.

3.2.7.4.2 SGMII Support

The X710-TM4 supports 1 Gb/s and 100 Mb/s operation using the SGMII protocol over the KX electrical interface (AC coupling, no source synchronous Tx clock, etc.).



3.2.7.4.2.1 SGMII Overview

SGMII interface supported by the X710-TM4 enables operation at 1 Gb/s over two differential, controlled impedance pairs of traces (one pair for transmit, one pair for receive). When operating in SGMII, the MAUI interface uses the 1000BASE-X PCS and PMA as defined in IEEE Std. 802.3-2018 Clause 36 and the 1000BASE-KX PMD as defined in IEEE Std. 802.3ap Clause 70 or the 1000BASE-BX as defined in the PCIMG 3.1 standard. In SGMII operating mode, the MAUI interface can support data rates of 1 Gb/s and 100 Mb/s. The X710-TM4 also supports IEEE Std. 802.3az EEE operation when configured to operate in SGMII mode (see [Section 5.3.1.7.2](#) for further information).

SGMII, supported by the X710-TM4, is a full-duplex interface that uses a single serial differential link in each direction to achieve 1 Gb/s data throughput. Each serial link operates at 1.25 GBaud to accommodate both data and the overhead associated with 8B/10B coding. The self-clocked nature eliminates skew concerns between clock and data.

SGMII control information, as listed in [Table 3-29](#) is transferred from the PHY to the MAC to signal change of link speed (100 Mb/s or 1 Gb/s). This is achieved by using the auto-negotiation functionality defined in Clause 37 of the IEEE Specification 802.3z. Instead of the ability advertisement, the PHY sends the control information via its tx_config_reg[15:0] as listed in [Table 3-29](#) each time the link speed information changes. Upon receiving control information, the MAC acknowledges the update of the control information by asserting bit 14 of its tx_config_reg[15:0] as listed in [Table 3-29](#). Compared to the definition in IEEE Std. 802.3-2018 clause 37, the link_timer inside the auto-negotiation has been changed from 10 ms to 1.6 ms to ensure a prompt update of the link status.

Table 3-29. SGMII Link Control Information

Bit Number	TX_CONFIG_REG[15:0] Sent From PHY to MAC	TX_CONFIG_REG[15:0] Sent From MAC to PHY
15	Link: 1b = link up, 0b = link down	0b = Reserved for future use.
14	Reserved for auto-negotiation acknowledge as specified in 802.3z	1b.
13	0b: Reserved for future use	0b = Reserved for future use.
12	Duplex mode: 1b = full duplex, 0b = half duplex	0b = Reserved for future use.
11:10	Speed: Bit 11, 10: 11b = Reserved. 10b = 1000 Mb/s: 1000BASE-TX. 01b = 100 Mb/s: 100BASE-TX. 00b = 10 Mb/s: 10BASE-T (not supported).	00b = Reserved for future use.
9	EEE: 1b = EEE is supported for 100BASE-TX and 1GBASE-T operation 0b = EEE is not supported.	0b = Reserved for future use.
8:1	0x0 = Reserved for future use.	0x0= Reserved for future use.
0	1b.	1b.

When operating in 100 Mb/s, the SGMII interface elongates the frame by replicating each frame byte 10 times for 100 Mb/s. This frame elongation takes place above the 802.3z PCS layer, thus the start frame delimiter only appears once per frame. Note that the 802.3z PCS layer might remove the first byte of the elongated frame. An example of a 100 Mb/s elongated frame can be seen in [Figure 3-11](#).

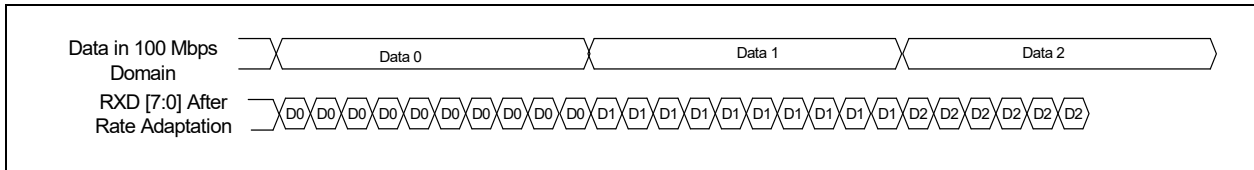


Figure 3-11. Data Sampling in 100 Mb/s Mode



3.2.8 PF-to-Lane Mapping

3.2.8.1 PF-to-Physical Lane Mappings (Supported Configurations)

Table 3-30 and Table 3-31 list port and possible port-to-physical lane configurations supported by the X710-TM4 for various PHY interfaces for single, dual and quad port configurations.

The Min SKU column in Table 3-31 indicates the supported SKUs:

- Dual 10G — X710-AT2
- Quad 10G — X710-TM4

Table 3-30. Port Configurations

Configuration #					
Port	Speed	Cfg ID 14.0 or 14.01	Cfg ID 14.0 or 14.2	Cfg ID 14.1 or 14.11	Cfg ID 14.3 or 14.4
0	10G	2 x 10G Base-T	2 x 10G Base-T	2 x 10G Base-T	2 x 10G Base-T
1					
2	10G	Backplane	2 x 10G Base-T ¹	2 x SFP	
3					
SKU		X710-TM4	X710-TM4	X710-TM4	X710-AT2

1. External PHY ports 2 and 3.

Table 3-31. Possible Port to Physical Lane Configurations for Various PHY Interfaces

Cfg ID	Min SKU	Application	Interface Protocol	Physical lane: Group A				Physical lane: Group B			
				L0	L1	L2	L3	L0	L1	L2	L3
14.0	Quad 10G	Quad port DP external PHY and DP KR	KR, KX, AN	PF0.L0		PF1.L0		PF2.L0		PF3.L0	
14.01	Quad 10G	Quad port DP external PHY and DP KR	PF0-1: SFI, SGMII PF2-3: KR, KX, AN	PF0.L0		PF1.L0		PF2.L0		PF3.L0	
14.1	Quad 10G	Quad port DP external PHY and DP SFI	PF0-1: KR,KX,AN PF2-3: SFI, SGMII	PF0.L0		PF1.L0		PF2.L0		PF3.L0	
14.11	Quad 10G	Quad port DP external PHY and DP SFI	SFI, SGMII	PF0.L0		PF1.L0		PF2.L0		PF3.L0	
14.2	Quad 10G	Quad port external PHY	KR, KX, AN	PF0.L0		PF1.L0		PF2.L0		PF3.L0	
14.3	Dual 10G	Dual port external PHY	KR, KX, AN	PF0.L0		PF1.L0					
14.4	Dual 10G	Dual port external PHY	SFI, SGMII	PF0.L0		PF1.L0					



3.2.9 Supported Media Types

The following table lists the supported SFP and SFP+ setups.

Table 3-32. Supported Media Types

Speed	Media	Description
10G	SFP+ SR / LR single-speed (10G) and multi-speed (1G / 10G) optical modules	<ul style="list-style-type: none">• These modules are identified based on the compliance code field.
	SFP+ DA twin-ax cables	<ul style="list-style-type: none">•
	SFP+ AoC's (Active optical Cables)	<ul style="list-style-type: none">• These modules are identified based on the compliance code field.• AoC cables are enabled each time the device is enabled for 10G-SR <p>Note: Only limiting initialization cables are supported.</p>
	Breakout DA twin-ax cables or Active Copper cables	<ul style="list-style-type: none">• These modules are identified based on the compliance code field.



3.2.10 Link Management Interfaces

3.2.10.1 Link Management Interfaces

The X710-TM4 supports either MDIO or I²C interfaces for control plane connection between the MAC (master side) and external PHY devices. The MDIO or I²C interfaces enable both MAC and firmware access to the PHY for monitoring and controlling of the PHYs functionality. The X710-TM4 MDIO is compliant with the IEEE Std 802.3 Clause 45 as well as IEEE Std 802.3 Clause 22 frame formats and register address space for accessing legacy PHY devices. The X710-TM4 I²C is compliant with the I²C bus specification.

The X710-TM4 supports up to two management interfaces (one per port) to control external PHY devices. Depending on the PHY type to manage, the management interface can be configured for either MDIO or 2-wire management interface. The configuration is done through control bits in the NVM.

[Section 3.2.10.2](#) includes the details on MDIO interface. [Section 3.2.10.4](#) includes the details on I²C interface.

In order to manage multi-port PHYs, an I²C/MDIO interface can be configured to control a dual port PHY. The PHY configuration registers for each port are mapped into the respective I²C/MDIO address space. The X710-TM4 provides hardware acceleration of MDIO accesses over the 2-wire management interface. The software device driver manages the external devices using the admin commands (see Link Configuration admin commands). The software device driver does not have direct access to the MDIO bus except for diagnostic purposes. Firmware performs direct access to MDIO interface for reading and writing to the PHY device as described in the sections that follow.

[Figure 3-12](#) shows a basic example of connectivity between the MAC and an external PHY/module.

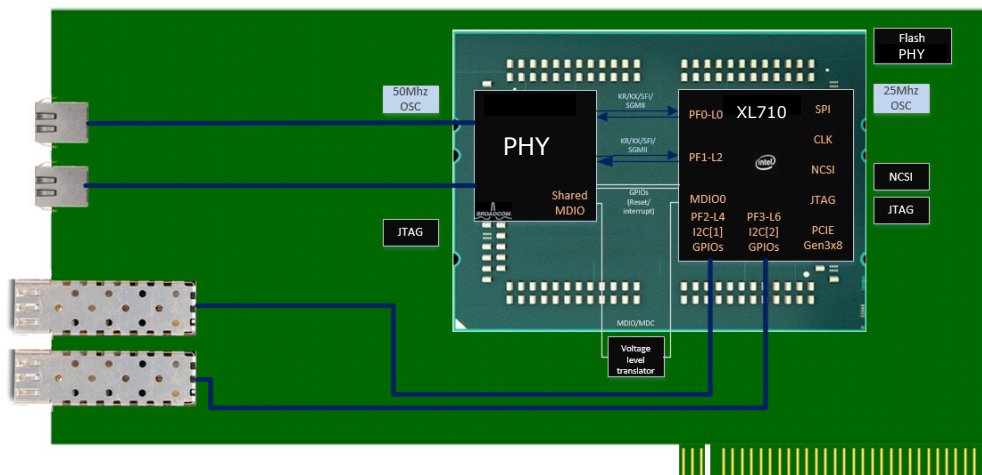


Figure 3-12. PHY/Module Connectivity (External PHY Connection Shown)

3.2.10.2 MDIO Interface

The MDIO interface is a simple 2-wire serial interface between MAC and PHY and is used to access Control and Status registers inside the PHY. The interface is implemented using two I/Os (3.3V I/Os for the MAC and 1.8V I/Os for the internal PHY):

1. MDC — MDIO-interface clock signal driven by a MAC (STA) device.
2. MDIO — Read/write data between MAC and PHY.

3.2.10.2.1 MDIO Timing Relationship to MDC

The MDC clock toggles during a read/write operation at a fixed clock frequency of 2.441 MHz.

MDIO is a bidirectional signal that can be sourced by the Station Management Entity (STA) or the PHY. When the STA sources the MDIO signal, the STA must provide a minimum of 10 ns of setup time and a minimum of 10 ns of hold time referenced to the rising edge of MDC, as shown in [Figure 3-13](#) (measured at the MII connector).

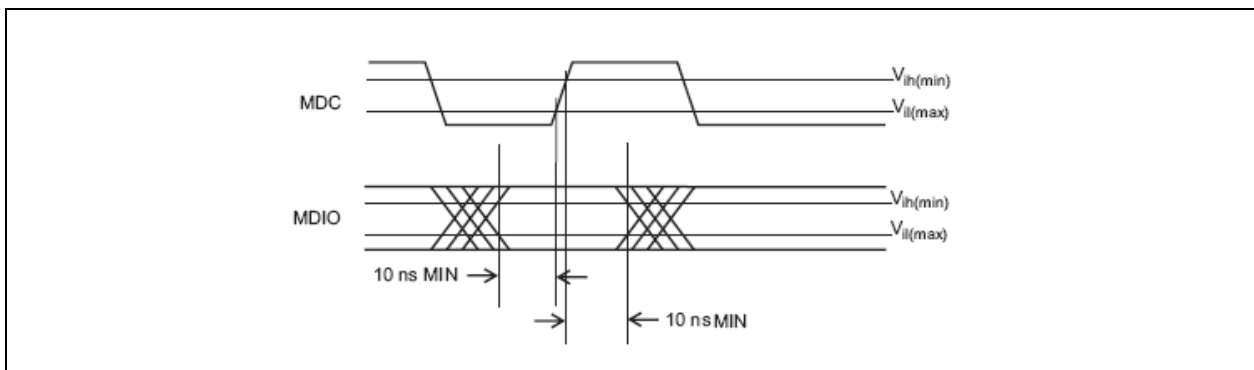


Figure 3-13. MDIO Timing Sourced by the MAC

When the MDIO signal is sourced by the PHY, it is sampled by the MAC (STA) synchronously with respect to the rising edge of MDC. The clock to output delay from the PHY, as measured at the MII connector, must be a minimum of 0 ns, and a maximum of 300 ns, as shown in [Figure 3-14](#).

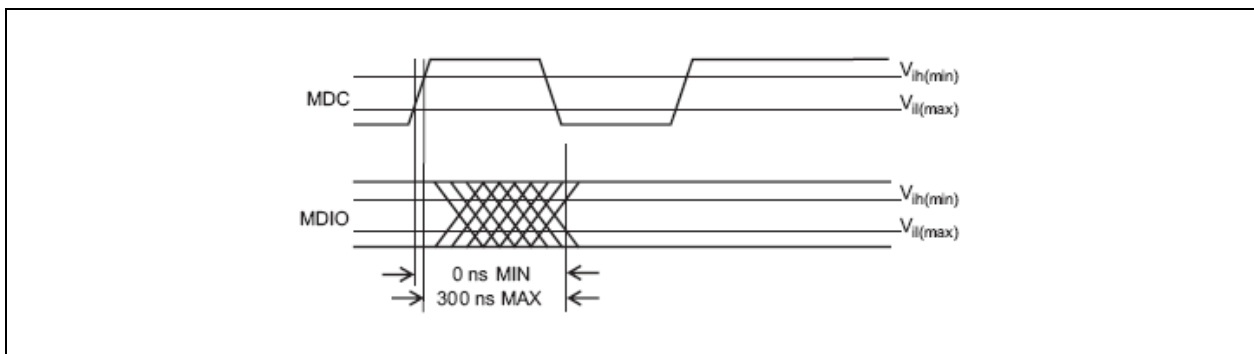


Figure 3-14. MDIO Timing Sourced by the PHY



3.2.10.2.2 IEEE Std 802.3 Clause 22 and Clause 45 Differences

IEEE Std 802.3 Clause 45 provides the ability to access additional device registers while still retaining logical compatibility with interface defined in Clause 22. Clause 22 specifies the MDIO frame format and uses an ST code of 01 to access registers. In Clause 45, additional registers are added to the address space by defining MDIO frames that use a ST code of 00.

Clause 45 (MDIO interface) major concepts:

- a. Preserve management frame structure defined in IEEE Std. 802.3-2018 Clause 22.
- b. Define mechanism to address more registers than specified in IEEE Std. 802.3-2018 Clause 22.
- c. Define ST and OP codes to identify and control the extended access functions.

3.2.10.2.3 MDIO Management Frame Structure

The MDIO interface frame structure defined in IEEE Std. 802.3-2018 Clause 22 and Clause 45 are compatible so that the two systems supporting different formats can co-exist on the same MDIO bus. The X710-TM4 supports both frame structures to enable interfacing PHYs that support either protocol.

The basic frame format as defined in IEEE Std. 802.3-2018 Clause 22 can optionally be used for accessing legacy PHY registers is listed in [Table 3-33](#).

Table 3-33. Clause 22 Basic MDIO Frame Format

	Management Frame Fields							
Frame	Pre	ST	OP	PRTAD	REGAD	TA	Data	Idle
Read	1...1	01	10	PPPPP	RRRRR	Z0	DDDDDDDDDDDDDDDD	Z
Write	1...1	01	01	PPPPP	RRRRR	10	DDDDDDDDDDDDDDDD	Z

The MDIO interface defined in Clause 45 uses indirect addressing to create an extended address space enabling access to a large number of registers within each MDIO Managed Device (MMD). The MDIO management frame format is listed in [Table 3-34](#).

Table 3-34. Clause 45 Indirect Addressing MDIO Frame Format

	Management Frame Fields							
Frame	Pre	ST	OP	PRTAD	DEVAD	TA	Address / Data	Idle
Address	1...1	00	00	PPPPP	EEEEEE	10	AAAAAAAAAAAAAAAA	Z
Write	1...1	00	01	PPPPP	EEEEEE	10	DDDDDDDDDDDDDDDD	Z
Read	1...1	00	11	PPPPP	EEEEEE	Z0	DDDDDDDDDDDDDDDD	Z
Post-Read Increment Address	1...1	00	10	PPPPP	EEEEEE	Z0	DDDDDDDDDDDDDDDD	Z



The device address should be set according to the target module to be accessed:

Target Area	Target Device	Device Address
Top Level	Top Level	0x1E
Line Side (BASE-T)	10GBASE-T PMA/PMD	0x1
	10GBASE-T PCS	0x3
	Auto-negotiation	0x7
	100/1000BASE-T PCS/PMA/PMD	0x7 (FFE0-FFFF)
Host (MAC) Side	XFI_M	0x3
	SGMII (XGMII)	0x7 (FFE0-FFFF)

To support clause 45 indirect addressing each MMD (PHY — MDIO managed device) implements a 16-bit address register that stores the address of the register to be accessed by data transaction frames. The address register must be overwritten by address frames. At power up or device reset, the contents of the address register are undefined. Write, read, and post-read-increment-address frames must access the register whose address is stored in the address register. Write and read frames must not modify the contents of the address register. Upon receiving a post-read-increment-address frame and having completed the read operation, the MMD increments the Address register by one (up to a value of 0xFFFF). Each MMD supported implements a separate address register, so that the MMD's address registers operate independently of one another.

Idle Condition (IDLE) — The IDLE condition on MDIO is a high-impedance state. All three state drivers must be disabled and the PHY's pull-up resistor pulls the MDIO line to a logic one.

Preamble (PRE) — At the beginning of each transaction, the station management entity must send a sequence of 32 contiguous consecutive one bits on MDIO with 32 corresponding cycles on MDC to provide the PHY with a pattern that it can use to establish synchronization. A PHY must observe a sequence of 32 contiguous consecutive one bits on MDIO with 32 corresponding cycles on MDC before it responds to any transaction.

Start of Frame (ST) — The ST is indicated by:

- <00> pattern for clause 45 compatible frames for indirect access cycles.
- <01> pattern for clause 22 compatible frames for direct access cycles.

These patterns ensure a transition from the default value of one on the MDIO signal, and identifies the start of frame.

Operation Code (OP) — The OP field indicates the type of transaction being performed by the frame.

For Clause 45 compatible frames:

- A <00> pattern indicates that the frame payload contains the address of the register to access.
- A <01> pattern indicates that the frame payload contains data to be written to the register whose address was provided in the previous address frame.
- A <11> pattern indicates that the frame is an indirect read operation.
- A <10> pattern indicates that the frame is an indirect post-read-increment-address operation.

For Clause 22 compatible frames:

- A <10> pattern indicates a direct read transaction from a register.
- A <01> pattern indicates a direct write transaction to a register.



Port Address (PRTAD) — The PRTAD is five bits, allowing 32 unique PHY port addresses. The first *PRTAD* bit to be transmitted and received is the MSB of the address. A station management entity must have prior knowledge of the appropriate port address for each port to which it is attached, whether connected to a single port or to multiple ports.

Device Address (DEVAD) — The DEVAD is five bits, allowing 32 unique MMDs per port. The first *DEVAD* bit transmitted and received is the MSB of the address. This field is relevant only in clause 45 compatible frames (ST=<00>).

Register Address (REGAD) — The REGAD is five bits, allowing 32 individual registers to be addressed within each PHY. The first *REGAD* bit transmitted and received is the MSB of the address. This field is relevant only in clause 22 compatible frames (ST=<01>).

Turnaround (TA) — The TA time is a 2-bit time spacing between the *DEVAD* field and the *Data* field of a management frame. This is to avoid contention during a read transaction. For a read or post-read-increment-address transaction, both the STA and the PHY must remain in a high-impedance state for the first bit time of the TA. The PHY must drive a zero bit during the second bit time of the TA of a read or post read-increment-address transaction. During a write or address transaction, the STA must drive a one bit for the first bit time of the TA and a zero bit for the second bit time of the TA. [Figure 3-15](#) shows the behavior of the MDIO signal during the *TA* field of a read transaction.

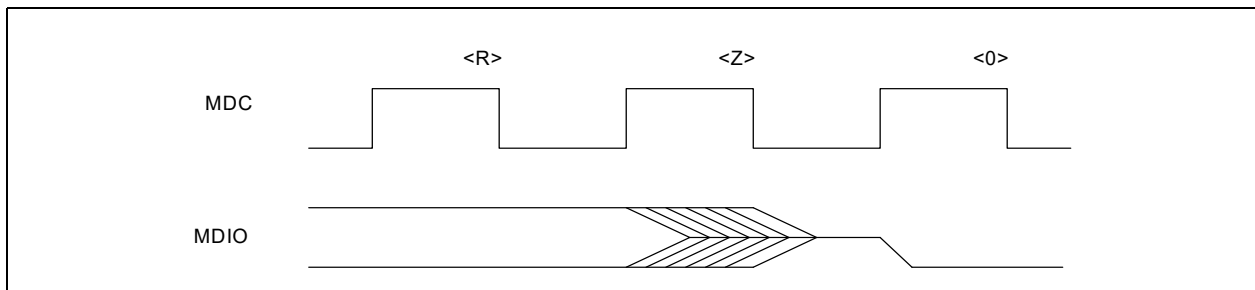


Figure 3-15. Behavior of MDIO During TA Field of a Read Transaction

- Clause 45 compatible frames have 16-bit address/data fields. For an auto-negotiation address cycle, it contains the address of the register to be accessed on the next cycle. For the data cycle of a write frame, the field contains the data to be written to the register. For a read or post-read-increment-address frame, the field contains the contents of the register. The first bit transmitted and received must be bit 15.
- Clause 22 compatible frames have 16-bit data fields. The first data bit transmitted and received must be bit 15 of the register being addressed.



3.2.10.2.4 MDIO Direct Access

The software device driver manages the PHYs using the Admin commands (See Link Configuration admin commands). The device driver does not have direct access to the MDIO bus except for diagnostic purposes. Firmware performs direct access to the MDIO interface for reading and writing to the PHYs as described in the following paragraphs.

The MDI is accessed through registers `GLGEN_MSCA` and `GLGEN_MSRWD`. A single management frame is sent by setting bit `GLGEN_MSCA.MDICMD` to 1b after programming the appropriate fields in the `MSCA` and `MSRWD` registers. The `GLGEN_MSCA.MDICMD` bit is auto cleared after the read or write transaction completes. To execute Clause 22 format write operations, the following steps should be done:

1. Data to be written is programmed in field `GLGEN_MSRWD.MDIWRDATA`.
2. Register `GLGEN_MSCA` is initialized with the appropriate control information (start, code, etc.) with bit `GLGEN_MSCA.MDICMD` set to 1b.
3. Wait for bit `GLGEN_MSCA.MDICMD` to reset to 0b when indicating that the transaction on the MDIO interface is complete.

The steps for Clause 22 format read operations are identical to the write operation except that the data in field `GLGEN_MSRWD.MDIWRDATA` is ignored and the data read from the external device is stored in register field `GLGEN_MSRWD.MDIRDDATA` bits. Clause 45 format read/write operations must be performed in two steps. The address portion of the pair of frames is sent by setting register field `GLGEN_MSCA.MDIADD` to the desired address, field `GLGEN_MSCA.STCODE` to 00b (start code that identifies Clause 45 format), and register field `GLGEN_MSCA.OPCODE` to 00b (Clause 45 address register write operation). A second data frame must be sent after the address frame completes. This second frame executes the write or read operation to the address specified in the PHY address register.

3.2.10.3 I²C - 2-wire Management Interface Engine

The I²C interface operates via the `GLGEN_I2CCMD` and `GLGEN_I2CPARAMS` register set. Software should not write to this register set when the interface is allocated to the firmware.

The I²C interface can be used in two methods, a hardware based access, where the device initiates a transaction following a software device driver request via the `GLGEN_I2CCMD` register (Section 10.2.1.1.24) or a software controlled bit banging using the `GLGEN_I2CPARAMS` register (Section 10.2.1.1.25).

3.2.10.3.1 Hardware-based I²C access

The following flows should be used to access an I²C register.

As part of device initialization, or anytime before the actual access, the following parameters should be set:

- `GLGEN_I2CPARAMS.PHYADD` - the address of the device to access.
- `GLGEN_I2CPARAMS.ACCESS_WIDTH` - the width of the data to read or write (byte or word).

Note: The `GLGEN_I2CPARAMS` register should not be modified during an I²C transaction.

To execute a write access, the following steps should be done:

1. Check that register is ready: Poll `GLGEN_I2CCMD.R` bit until it is read as 1b.



2. Command — The `GLGEN_I2CCMD` register is initialized with the appropriate PHY register address in `REGADD` field, the data to write in the `DATA` field and the operation (write) to the `OP` field (0b).
 - a. If an interrupt is required, set the `GLGEN_I2CCMD.I` field
3. Check that command is done: Poll `GLGEN_I2CCMD.R` bit until it is read as 1b.
 - a. Check that no error is indicated in the `GLGEN_I2CCMD.E` field.

To execute a read access, the following steps should be done:

1. Check that register is ready: Poll `GLGEN_I2CCMD.R` bit until it is read as 1b.
2. Command — The `GLGEN_I2CCMD` register is initialized with the appropriate PHY register address in `REGADD` field, and the operation (read) to the `OP` field (1b).
 - a. If an interrupt is required, set the `GLGEN_I2CCMD.I` field
3. Check that command is done: Poll `GLGEN_I2CCMD.R` bit until it is read as 1b.
 - a. Check that no error is indicated in the `GLGEN_I2CCMD.E` field.
4. Read the data returned from the `GLGEN_I2CCMD.DATA` field. If a byte access is done (`GLGEN_I2CPARAMS.ACCESS_WIDTH = 0`), only `DATA[7:0]` is valid.

See [Section 3.2.10.3.3](#) for the I²C commands supported when using the built in read and write commands. All the transactions uses a clock of 100 KHz. When using the bit bang method any command can be given to the I²C device.

3.2.10.3.2 Bit Bang-based I²C Access

In this mode, the software device driver controls the I²C interface directly using the `GLGEN_I2CPARAMS` register according to the following table:

Pad	Field Controlling Output Value	Field Reflecting Input Value	Field Controlling Output Enable Value ¹
SDPx_2 (i ² c clock)	CLK_OUT	CLK_IN	CLK_OE_N
SDPx_3 (i ² c data)	DATA_OUT	DATA_IN	DATA_OE_N

1. 0b = Pad is output. 1b = Pad is input.

3.2.10.3.3 Supported Commands

Note: The following gray columns denote cycles driven by the I²C device. White columns denote cycles driven by the X710-TM4.

When a word read command (`GLGEN_I2CPARAMS.ACCESS_WIDTH = 1b`, `GLGEN_I2CCMD.OP = 1b`) is given the following sequence is done by the X710-TM4:

Table 3.1. I²C Read Transaction - Dummy Write

1	7	1	1	8	1	
S	Device Address	Wr	A	Register Address	A	
	From <code>GLGEN_I2CCMD.PHYADD</code>	0	0	From <code>GLGEN_I2CCMD.REGADD</code>	0	



Table 3.2. I²C Read Transaction - Word Read

1	7	1	1	8	1	8	1	1
S	Device Address	Rd	A	Data	A	Data	A	P
	From GLGEN_I2CPARAMS.PHYADD	1	0	Stored in GLGEN_I2CCMD.DATA[7:0]	0	Stored in GLGEN_I2CCMD.DATA[15:8]	0	

When a Byte Read command (*I2CPARAMS.ACCESS_WIDTH* = 0b, *I2CCMD.OP* =1b) is given the following sequence is done by the X710-TM4:

Table 3.3. I²C Read Transaction - Dummy Write

1	7	1	1	8	1
S	Device Address	Wr	A	Register Address	A
	From GLGEN_I2CPARAMS.PHYADD	0	0	From GLGEN_I2CCMD.REGADD	0

Table 3.4. I²C Read Transaction - Byte Read

1	7	1	1	8	1	1
S	Device Address	Rd	A	Data	A	P
	From GLGEN_I2CPARAMS.PHYADD	1	0	Stored in GLGEN_I2CCMD.DATA[7:0]	0	

When a word write command (*I2CPARAMS.ACCESS_WIDTH* = 1b, *GLGEN_I2CCMD.OP* =0b) is given the following sequence is done by the X710-TM4:

Table 3.5. I²C Write Transaction - Word Write

1	7	1	8	1	8	1	8	1	1
S	Device Address	Wr	Register Address	A	Data	A	Data	A	P
	From GLGEN_I2CPARAMS.PHYADD	0	From GLGEN_I2CCMD.REGADD	0	From in GLGEN_I2CCMD.DATA[7:0]	0	From in GLGEN_I2CCMD.DATA[15:8]	0	

When a Byte Write command (*I2CPARAMS.ACCESS_WIDTH* = 0b, *I2CCMD.OP* =0b) is given the following sequence is done by the X710-TM4:

Table 3.6. I²C Write Transaction - Byte Write

1	7	1	8	1	8	1	1
S	Device Address	Wr	Register Address	A	Data	A	P
	From GLGEN_I2CPARAMS.PHYADD	0	From GLGEN_I2CCMD.REGADD	0	From in GLGEN_I2CCMD.DATA[7:0]	0	

3.2.10.4 Link Management Topologies

The X710-TM4 supports MDIO and 2-wire management interface (I²C) for connectivity to external modules, re-timers and PHYs. For example, SFP optical and direct attached copper PHYs.



The X710-TM4 supports up to two management interfaces (one per port) to control external PHY devices. Depending on the PHY type to manage these can be configured for either MDIO or 2-wire management interface (see [Section 3.2.10.2](#) for details on MDIO interface and [Section 3.2.10.4](#) for details on I²C interface). The configuration is done through control bits in the NVM.

[Figure 1-3](#) and [Figure 3-12](#) show the connectivity between the MAC and an external module.

The X710-TM4 provides hardware acceleration of I²C accesses over the 2-wire management interface. The software device driver manages the external modules using the Admin commands (see Link Configuration admin commands). The software device driver does not have direct access to the I²C bus except for diagnostic purposes. Firmware performs direct access to I²C interface for reading and writing to the PHY modules.

The I²C bus is accessed through registers `GLGEN_I2CCMD` and `GLGEN_I2CPARAMS`. I²C accesses through the hardware controller can be performed through the `GLGEN_I2CCMD` register. I²C accesses can also be performed using bit banging through `GLGEN_I2CPARAMS` register if needed. Firmware can execute I²C write/read operations by writing to the I²C command register. The status of I²C cycle completion can be performed by reading the `GLGEN_I2CCMD` register. Refer to the steps that follow to appropriately set the fields:

1. I²C register address is placed in `GLGEN_I2CCMD.DEVADD` field.
2. PHY address is placed in `GLGEN_I2CCMD.PHYADD` field (typically this is 0xA0 or 0xA2 for modules).
3. Command for read or write operation is placed in the `GLGEN_I2CCMD.OP` field.
4. For a write operation, data to be written is placed in the `GLGEN_I2CCMD.DATA` field.
5. Successful completion of a write or read operation is indicated by the X710-TM4 through *Ready* bit (`GLGEN_I2CCMD.R`). A read error is indicated if `GLGEN_I2CCMD.E` bit is set along with `GLGEN_I2CCMD.R`.
6. For a read operation, data can be read from `GLGEN_I2CCMD.DATA` field when the *Ready* (`GLGEN_I2CCMD.R`) bit is set.
7. A reset sequence can be sent to the I²C bus before the actual write/read operation when `GLGEN_I2CCMD.Reset` bit is set.

3.2.10.5 External PHY Management Connectivity

The X710-TM4 has designated 22 GPIO pins as SDPs for use with ports 2 through 3, as listed in [Section 2.2.6](#). This section describes the use of SDP pins for the management of external PHYs and optical/copper modules.

Firmware controls the SDP pins when used for specific hardware functions for PHY management. The software device driver uses admin commands (see Link Configuration admin commands) to configure and manage the PHYs. Note that the software device driver does not directly access the SDP pins except for diagnostic purposes.

[Table 3.3](#) lists the recommended configurations for connecting The X710-TM4's SDPs, MDIO/I²C pins to an external PHY or optical/copper module. The configurations include connectivity to SFP/SFP+ modules and 10GBASE-T PHYs. If mapping of these SDP pins to a specific hardware function is not required then the pins can be used as GPIOs.

Any of the SDP pins configured as an input for dedicated hardware functions can be configured to trigger an interrupt to the firmware. The firmware in turn reads the appropriate PHY status registers (through MDC or I²C interfaces) and might post an event and provide necessary status information to the software device driver through admin commands (see Link Configuration admin commands).



Table 3-35. SDP, MDIO and I²C Ports Usage for SFP+ Modules Example

Port Number	Pin Name	2x 10GBASE-T Only	2x 10GBASE-T and 2x SFP+	4x 10GBASE-T
Port 0	SDP0_0	P[0:1] BASE-T PHY RST (needs level translation)	P[0:1] BASE-T PHY RST (needs level translation) ¹	P[0:1] BASE-T PHY RST (needs level translation) ¹
	SDP0_1	P0 BASE-T PHY INT	P0 BASE-T PHY INT	P0 BASE-T PHY INT
	SDP0_2	P1 BASE-T PHY INT	P1 BASE-T PHY INT	P1 BASE-T PHY INT
	SDP0_3	Not used	P2 SFP+ TX_DIS	Not used
	MDIO0_SDA0	P[0:1] BASE-T PHY MDIO (needs level translation)	P[0:1] BASE-T PHY MDIO (needs level translation)	P[0:1] BASE-T PHY MDIO (needs level translation)
	MDC0_SCL0	P[0:1] BASE-T PHY MDC (needs level translation)	P[0:1] BASE-T PHY MDC (needs level translation)	P[0:1] BASE-T PHY MDC (needs level translation)
Port 1	SDP1_0	Not used	P2 SFP+ RX_LOS	Not used
	SDP1_1	Not used	P2 SFP+ TX_FAULT	P[2:3] BASE-T PHY RST ¹ (needs level translation)
	SDP1_2	Not used	P2 SFP+ MOD_ABS	P2 BASE-T PHY INT ¹
	SDP0_3	Not used	P2 SFP+ RS0_RS1	P3 BASE-T PHY INT ¹
	MDIO1_SDA1	Not used	P2 SFP+ SDA	Not used
	MDC1_SCL1	Not used	P2 SFP+ SCL	Not used
Port 2	SDP2_0	Not used	Not used	Not used
	SDP2_1	Not used	Not used	Not used
	SDP2_2	Not used	Not used	Not used
	SDP2_3	Not used	Port 3 SFP+ TX_DIS	Not used
	MDIO2_SDA2	Not used	Not used	P[2:3] BASE-T PHY MDIO (needs level translation)
	MDC2_SCL2	Not used	Not used	P[2:3] BASE-T PHY MDC (needs level translation)
Port 3	SPD3_0	Not used	P3 SFP+ RX_LOS	Not used
	SPD3_1	Not used	P3 SFP+ TX_FAULT	Not used
	SPD3_2	Not used	P3 SFP+ MOD_ABS	Not used
	SDP3_3	Not used	P3 SFP+ RS0_RS1	Not used
	MDIO3_SDA3	Not used	P3 SFP+ SDA	Not used
	MDC3_SCL3	Not used	P3 SFP+ SCL	Not used
LED	LED0_0	P0 Activity LED	P0 Activity LED	P0 Activity LED
	LED0_1	P0 10G Link LED	P0 10G Link LED	P0 10G Link LED
	LED1_0	Not used	P2 Activity LED	P2 Activity LED
	LED1_1	Not used	P2 10G Link LED	P2 10G Link LED
	LED2_0	P1 Activity LED	P1 Activity LED	P1 Activity LED
	LED2_1	P1 10G Link LED	P1 10G Link LED	P1 10G Link LED
	LED3_0	Not used	P3 Activity LED	P3 Activity LED
	LED3_1	Not used	P3 10G Link LED	P3 10G Link LED



Table 3-35. SDP, MDIO and I²C Ports Usage for SFP+ Modules Example

Port Number	Pin Name	2x 10GBASE-T Only	2x 10GBASE-T and 2x SFP+	4x 10GBASE-T
GPIO	GPIO_0	P0 <10G Link LED	P0 <10G Link LED	P0 <10G Link LED
	GPIO_1	Not used	P2 <10G Link LED	P2 <10G Link LED
	GPIO_2	P1 <10G Link LED	P1 <10G Link LED	P1 <10G Link LED
	GPIO_3	Not used	P3 <10G Link LED	P3 <10G Link LED
	GPIO_4	Not used	Not used	Not used
	GPIO_5	POR_BYPASS	POR_BYPASS	POR_BYPASS

Table 3-36 lists additional information about the SDPs used for managing external PHYs and modules, their GPIO pin configuration and behavior during reset and power down state (D3) without management.

Table 3-36. SDP Assigned Signals/Pin Configuration

Signal	Description	GPIO Pin Configuration	Default Values at (Reset, D3 no WoL and no MNG) ¹
RX_LOS	When high, RX_LOS indicates the optical signal is below the relevant link mode standard.	SDP: Input (Interrupt)	Input, no change.
LASI, INTR_L	INTR_L or Link Alarm Status Interrupt (LASI) — when low, indicates possible module operational fault or a status critical to the host system.	SDP: Input (Interrupt)	Input, no change.
RS0/RS1 drive	Short-circuit protected.	SDP: Output	Output, autonomous high or tri-state with pull up.
RS0/RS1 sense	Directly connected input.	SDP: Input	Input, no change.
MOD_ABS	When low, indicates SFP module is present; when high, indicates that the module is absent.	SDP: Input (Interrupt)	Input, no change.
TX_DISABLE	When TX_DISABLE is asserted high, optical module transmitter is turned off.	SDP: Output	Output, no change. In order to minimize PHY power software should drive the SDP to high or set to input while populating a pull up.
TX_FAULT	When high, indicates that the module transmitter has detected a fault condition related to laser operation or safety.	SDP: Input (Interrupt)	Input, no change.
RESET	When high, the copper PHY is reset.	SDP: Output	Output, autonomous high or tri-state with pull up.
TX_DIS	When TX_DIS is asserted high, optical module transmitter is turned off.	SDP: Output	Output, autonomous high or tri-state with pull-up.
TX ON/OFF	1b = Transmitter on. 0b = Transmitter off.	SDP: Output	Output, autonomous low or tri-state with pull-down.
MOD_DET_N	Inverted mode detect.	SDP: Input (Interrupt)	Input, no change.
TS_SDPX	Time sync support pins, can be used as event in or event out.	According to programmed functionality	Tri-state during reset. No change in D3. External pull-up / pull-down as required by the system designer.



Table 3-36. SDP Assigned Signals/Pin Configuration

Signal	Description	GPIO Pin Configuration	Default Values at (Reset, D3 no WoL and no MNG) ¹
MOD_NR	When high, indicates that the module has detected a condition that renders transmitter and or receiver data invalid.	SDP: Input (Interrupt)	Input, no change.
ModSell	When Low, I ² C access to module is enabled, else I ² C access is disabled.	SDP: Output	Output, PHY software to drive the line low for access through I ² C interface.
FAN_Status	Optional health indication of the fan.	SDP: Input (Interrupt)	Input, no change.

1. Use GLGEN_GPIO_CTL.OUT_CTL to control the default output (tri-state or driven high) during reset or D3 power state.

3.2.10.5.1 Transceiver Module Support

The X710-TM4 MAUI interface with additional usage of low speed interface pins (SDP, I²C and MDIO I/Os) supports connection to transceiver modules compliant with the following Multi Source Agreements (MSAs):

- XENPAK — A cooperation agreement for 10 GbE transceiver package Rev 3.0
- X2 — A cooperation agreement for a small Versatile 10 GbE transceiver package Rev 2.0b
- XPAK — A cooperation agreement for a small form factor pluggable 10 GbE transceiver package Rev 2.2
- SFP+ — SFF-8431 Specifications for Enhanced 8.5 and 10 Gigabit Small Form Factor Pluggable Module SFP+ rev 1.0

3.2.10.5.2 X710-TM4’s SFP Connectivity Scheme

Refer to the *Intel® Ethernet Controller X710-TM4/AT2 Reference Schematics* for detail.

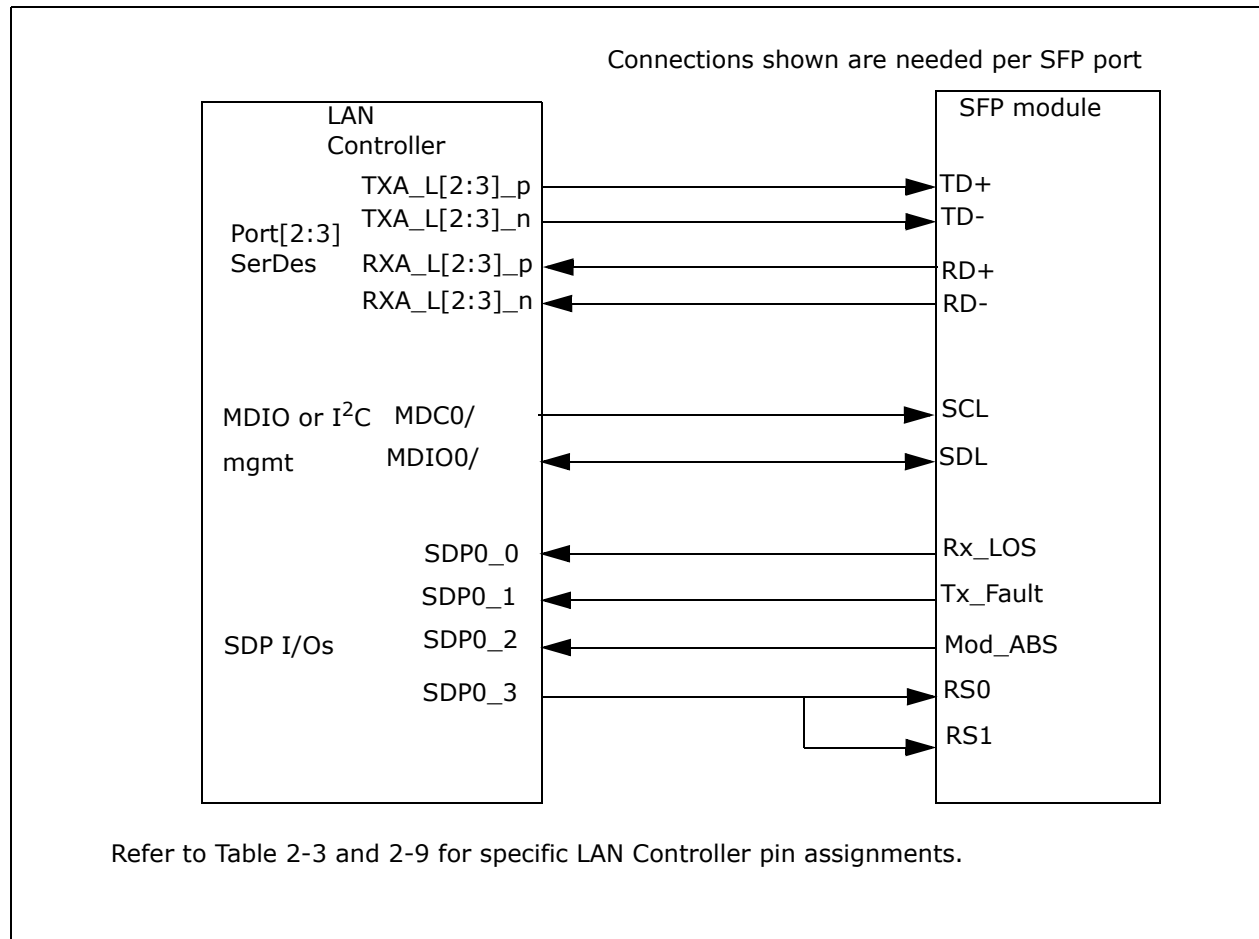


Figure 3-1. X710-TM4 SDP Connections to SFP Modules

3.2.11 Link Configuration

3.2.11.1 Firmware Link Management

In the X710-TM4, all link configurations are performed by firmware. At POR, firmware initializes the link and then provides a set of Admin commands, described in [Section 3.2.9](#), to allow for software device drivers’ modifications of the link parameters and configurations. In the X710-TM4, all access to the link configurations is performed using the Admin commands and software device drivers do not directly access the MAC, PHY or SDP settings. This is required in order to provide the software device driver with an API like interface to the MAC, PHY and SDP registers.



Link initialization by firmware includes reading the initial configuration from the NVM, reading the PHY type connected to the link and programming appropriate values to the MAC and PHY registers to bring up the link. There is L2 switch connected between the Ethernet port and the PHY. See the detailed flow in [Section 3.2.11.2](#) for more information.

The firmware also provides services to the software device driver by providing status of the link, modifying the link configuration, and re-initializing the link if requested by the software device driver.

3.2.11.2 Link Setup Flow

The X710-TM4's interface is configured at start up by firmware (before the software device driver loads) in the following manner:

1. Firmware reads default settings from the NVM per port.
2. Firmware reads GPIO, MDC/I²C settings preloaded by hardware from the NVM (see [Section 6.2.29](#)).
3. If link is setup for backplane connectivity:
 - Firmware programs the internal PHY registers and restarts the auto-negotiation process.
 - Firmware waits for link to be established.
 - Based on resulting speed and parameters (EEE,FC) firmware configures other blocks in the X710-TM4.
4. If link is setup for module connectivity:
 - Firmware verifies that the connected module is a qualified module (when enabled)
 - Based on module type, optical or direct attach, firmware configures the PCS registers, and the re-timer (when applicable).
 - Firmware restarts the link setup.
5. If link is setup for BASE-T PHY connectivity (external PHY):
 - Firmware verifies that the connected PHY is valid.
 - Firmware programs external PHY's registers
 - Firmware restarts the auto-negotiation process on the BASE-T interface.
 - When link is established, firmware programs the internal MAC-PHY interface according to the negotiated speed and parameters (EEE,FC) as well as other relevant blocks in the device.
6. When internal link is established, firmware generates a Link Status Event (LSE) if enabled.

Note: If link synchronization is not successful, firmware does not report link-up and continuously polls link indications while waiting for link to be established.

If module or PHY qualification is enabled and the connected device is not found in the qualified list, firmware stops the link setup process and reports event to software.

3.2.11.2.1 Software Link Management

At power up, firmware initializes the link based on the initial configurations loaded from the NVM. Following the initial setup, the software device driver might override any of the link settings; however, even though the software device driver is the owner of the link in SFP mode, it does not directly access the MAC or PHY registers and SDP pins controlling the PHYs instead all access to the link settings is performed through firmware using the Link Configuration admin commands.



3.2.11.2.2 MAC Link Setup and Auto-negotiation

Link speed and link characteristics can be determined through static configuration, parallel detection, auto-negotiation or forced operation for diagnostic purposes. The auto-negotiation processes is defined in IEEE Std. 802.3-2018 Clause 73 and IEEE Std. 802.3-2018 Clause 37 and enables detection of link speed and other parameters.

3.2.11.2.2.1 Auto-negotiation for Backplane Ethernet and Direct Attached Copper Cable Assembly (IEEE Std. 802.3-2018 Clause 73)

Auto-negotiation provides the link partners the capability to advertise and detect the modes of operation supported at the other end of the link, determine common abilities, and configure for joint operation.

Auto-negotiation for the backplane Ethernet is specified by IEEE Std. 802.3-2018, Clause 73.

Auto-negotiation defined in IEEE Clause 73 enables automatic link detection and setup for the following PHY: 10GBASE-KR

Auto-negotiation also includes support for parallel detection of 1000BASE-KX legacy links that don't implement Clause 73 auto-negotiation.

The X710-TM4 supports transmitting and receiving extended base page and next page auto-negotiation frames.

3.2.11.2.2.2 Next Page (NP) Support

NP support in the X710-TM4 is compliant with IEEE Clause 73.

The X710-TM4 acts as receiver of NP each time the link partner needs to transmit NP data through the KX auto-negotiation process.

The X710-TM4 supports transmitting a configurable NP. It transmits an NP each time the auto-negotiation arbitration state machine is required to go through the NP handshake.

3.2.11.2.2.3 Auto-negotiation Selection

Link speed and link characteristics can be determined through static configuration or auto-negotiation depending on the type of link being used. Firmware selects to enable or disable auto-negotiation according to the link mode as described in the following table:

Table 3-37. Auto-negotiation Selection

Link Mode	Auto-negotiation	Specification
Backplane (KX/KR)	Enable	IEEE Clause 73
External Optical Module (SFP)	Disable	N/A
1 GbE Optical or BASE-T SFP module	Enable	IEEE Clause 37
BASE-T	Enable	
SGMII		IEEE Clause 37



3.2.11.2.2.4 Link Establish vs. Fault Status

If remote fault is set, firmware waits a pre-defined time-out (5 seconds) before restarting the PCS (auto-negotiation state machine) without changing the PCS configuration (preserving all link parameters: PHY type, link speed, LFC, etc.).

Initiate link establish state machine that might change PHY settings only if the local fault (that exceed its time-out).

3.2.11.2.2.5 Auto-negotiation and Next Page Dilemma

This section describes the flow addressing non-specification compliant link partner that does not support next pages. Such scenario happened while attempting to establish link via AN with NP against a Pass Through Module (PTM) that supports KR-10G without NPs.

The following flow chart shows the Link Establishment State Machine (LESM) that overcome the previous case. LESM avoids an infinite wait for NPs (if not supported) by an attempt to establish link without a an NP dependent feature like EEE (this option is not activated if EEE is disabled by the NVM, software or manageability).

Table 3-38. LESM States

LESM State	Description
Initialization	Power up the PHY. Restart the PHY AN with all supported protocols, including next page dependent features.
Wait LP connected	Sample hardware AN state machine every ~30 ms. Wait until the link partner is connected and responding via AN protocol by verifying that hardware AN state exists in the ABILITY_DETECT states.
Try AN with NP	Power down the PHY for 0.5 seconds. Restart AN with all supported protocols, including next page dependent features (advertise EEE). Wait 2 seconds for AN to complete with MAC link up.
Try AN without NP	Power down the PHY for 0.5 seconds. Restart AN without next page dependent features. Wait 2 seconds for AN to complete with MAC link up.
Monitor MAC link	Restart the PHY AN hardware state machine every 5 seconds in case the MAC link is down with remote faults. The PCS configuration remains, thus keeping link parameters. PHY link is up. Restart firmware link state machine in case the MAC link is down with local fault for more than 5 seconds. PHY link is down.

Table 3-39. LESM Transitions

LESM Transition	Description
0	One of the following: 1. Power on. 2. New hardware setup configuration. 3. Firmware reset. 4. Host reconfigure link.
1	Initialization completed automatically.
2	Hardware AN state machine is in one of the ABILITY_DETECT states (LP does not respond to AN protocol).



Table 3-39. LESM Transitions

LESM Transition	Description
3	Hardware AN state machine exists the ABILITY DETECT states (LP is connected and responding to AN protocol).
4	AN completed successfully with MAC link is set up with EEE.
5	AN does not complete within two seconds.
6	AN completed successfully with MAC link is up without EEE.
7	AN does not complete within two seconds.
8	One of the following: 1. MAC link is up. 2. MAC link is down with remote fault set. MAC link is down with local fault set for less than two seconds.
9	Mac link is down with local fault for at least 2 seconds.

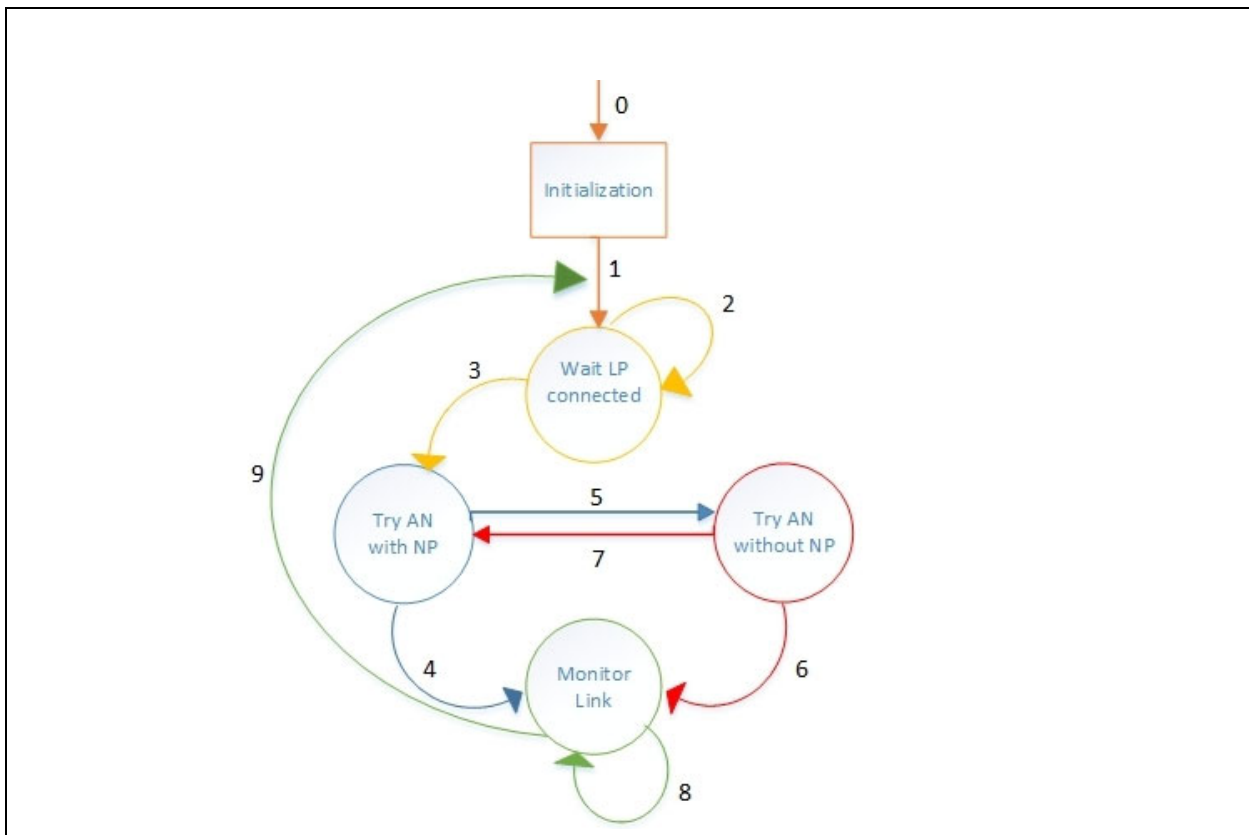


Figure 3-2. Link Power Management State Diagram



3.2.11.3 External PHY Power Mode Control

This section describes the flow that X710-TM4 executes controlling the power mode of external modules, and in addition the possible power modes of each external PHY/module (BASE-T PHYs). The flow should be executed in the following cases:

- Power on
- A module is plugged in

3.2.11.3.1 BASE-T PHY Modules

Low power mode is a way to place the disable the link and reduce power consumption.

3.2.11.3.1.1 BASE-T Power Modes

In order to set the power mode, X710-TM4 should use one of the following controls:

- The LPMode pin (SDP) if this is connected
- Power_override and Power_set bits (Address 0xA0, byte 93 bits 0,1)

3.2.11.3.1.2 BASE-T power Management Flow

1. If [(\langle system low power ability \rangle = high power) or (BASE-T PHY doesn't support low power)].
 - a. Set the PHY to high power mode -> link is enabled.
2. Else if (\langle system low power ability \rangle = low power).
 - a. Set the PHY to low power mode -> link i disabled.

3.2.12 Link Configuration Admin Commands

The X710-TM4 supports the following Admin commands for configuring and managing the link. Software should use the Admin commands to configure the link. This includes configuring the MAC and internal/external PHY devices. Firmware provides link configuration and status services to the software device driver based on these Admin commands.

Table 3-40. Link Configuration Admin Commands (0x06xx)

Command	Opcode	Description	Detailed Description
Set PHY Config	0x0601	Set various PHY configuration parameters on port.	Section 3.2.12.1.1
Set MAC Config	0x0603	Set various MAC configuration parameters on the port.	Section 3.2.12.1.2
Setup Link and Restart Auto-negotiation ¹	0x0605	Sets up the link and restarts link auto-negotiation. This operation could bring down the link. This command needs to be executed for other set link parameters to take effect on the link.	Section 3.2.12.1.3
Get PHY Abilities	0x0600	Get various PHY abilities supported on the port.	Section 3.2.12.1.4
Get Link Status	0x0607	Get link status of the port.	Section 3.2.12.1.5

**Table 3-40. Link Configuration Admin Commands (0x06xx)**

Command	Opcode	Description	Detailed Description
Link Status Event (Opcode: 0x0607)	0x0607	Firmware sends this asynchronous event notification to software when there is a change in status in any of the event causing conditions (such as link up/down or other link error conditions).	Section 3.2.12.1.6
Set Event Mask	0x0613	Sets event mask. Software can mask some or all of the link status event causing conditions.	Section 3.2.12.1.7
Set Loopback Modes	0x0618	Sets various MAC/PHY loopback modes of the link.	Section 3.2.12.1.8
Set PHY Register	0x0628	Write to internal or external PHY register	Section 3.2.12.1.9
Get PHY Register	0x0629	Read from internal or external PHY register	Section 3.2.12.1.10

1. In SFP mode, the setup link and restart auto-negotiation command needs to be executed by the software device driver in order for any other change in link parameters to take effect on the link. This operation could disrupt the link since the link state might toggle while the link is re-initialized with the new parameters.

3.2.12.1 Link Configuration Commands

This section provides a detailed description of the link configuration admin commands and its structure.

3.2.12.1.1 Set PHY Config

This command is used by the software device driver to set the various PHY configuration parameters supported on the port.

This is a Direct command. The set PHY command parameters data structure is placed in the descriptor.

Note: This command must be followed by the [Setup Link and Restart Auto-negotiation](#) command in order for any changes to the link parameters to actually take place.

Table 3-41. Set PHY Config Command (Opcode: 0x0601)

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0b, value is ignored.
Return Value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Set PHY Config	16-31	See Table 3-42	16-byte data structure that holds the set PHY Config command parameters listed in Table 3-42 .

[Table 3-42](#) lists the data structure of the set PHY config command parameters such as PHY type, PHY ID, speed ability, pause ability, etc.



Table 3-42. Set PHY Config Command Data Structure

Name	Bytes.Bits	Value	Remarks
PHY Type	0-3	PHY Type	PHY type supported on port. One bit per PHY type. The X710-TM4 might be capable of supporting multiple PHY types. The following parameter indicates the bit number. 00 = SGMII. 01 = 1000BASE-KX. 02 = Reserved. 03 = 10GBASE-KR. 04 = Reserved. 05 = Reserved. 06 = XFI. 07 = SFI. 08 = Reserved. 09 = Reserved. 10 = Reserved. 11 = Reserved. 17 = 100BASE-TX. 18 = 1000BASE-T. 19 = 10GBASE-T.
			20 = 10GBASE-SR (10 Gb/s SFP+ SR optical module). 21 = 10GBASE-LR (10 Gb/s SFP+ LR optical module). 22 = 10GBASE-SFP+Cu (direct attach copper). 23 = Reserved. 24 = Reserved. 25 = Reserved. 26 = Reserved. 27 = 1000BASE-SX. 28 = 1000BASE-LX. 29 = 1000BASE-T Optical Other bits - Reserved, must be zero.
			This parameter is used by the device driver to set the various PHY types' configuration parameters to be supported on the port. The port can be configured for a subset of the actual PHY types available on the port. The actual PHY types available are read by the device driver using Get PHY Abilities command. When auto-negotiation is enabled, the X710-TM4 negotiates and selects one of the PHY types enabled. When auto-negotiation is disabled, this field should enable only a single value and then the X710-TM4 is forced to operate in that selected mode.
Link Speed	4	Link Speed	Set link speed on port, only one operational speed is set by the device driver. Bit 4.0 = 2.5 Gb/s. Bit 4.1 = 100 Mb/s. Bit 4.2 = 1 Gb/s. Bit 4.3 = 10 Gb/s. Bit 4.4 = Reserved. Bit 4.5 = Reserved Bit 4.6 = Reserved. Bit 4.7 = 5 Gb/s. Other bits - Reserved, Must be zero. This parameter is used by the software device driver to set the operational link speed of the port. The X710-TM4 might have the ability to support multiple link speeds on the same port that can be found using the Get PHY ability command (see Section 3.2.12.1.4). One of the link speeds can be enabled due to the result of auto-negotiation. This command can be used by the software device driver to manually set the link speed when auto-negotiation is disabled.



Table 3-42. Set PHY Config Command Data Structure (Continued)

Name	Bytes.Bits	Value	Remarks
Pause Ability	5.0:5.1	Pause Ability	5.0 set to 1b to enable IEEE Std. 802.3x Tx link pause ability, or set to 0b to disable pause ability. 5.1 set to 1b to enable IEEE Std. 802.3x Rx link pause ability, or set to 0b to disable pause ability. Auto-negotiation might have be restarted for this configuration to take effect over the link. This parameter is used by the device driver to set the IEEE Std. 802.3x pause ability of the port. The X710-TM4's pause ability can be read by using the Get Link Status or Get PHY Abilities commands. The device driver might disable the IEEE Std. 802.3x link pause ability using this command. If the link is already up and configured, the device driver needs to restart auto-negotiation, so the updated pause ability could be advertised to the link partner in order for the setting to take effect on the link.
Low Power Ability	5.2	Low Power Mode	1: Low Power Mode 0: High Power Mode BASE-T: this is ignored if NVM loaded <Low Power Ability> = high power <ul style="list-style-type: none"> Firmware sets the power mode based on the setting of the first (lower power number) port in the group and ignore the settings of the other ports in the group.
Enable Link	5.3	Enable Link	Set to 1b to enable the link. Set to 0b to disable the link. The software device driver should not force link down when a port is being used for manageability or WoL.
Reserved	5.4	Reserved	
Enable Atomic Link Update	5.5	Enable Atomic Link Update	When this field is set to 1b, firmware automatically executes the Setup Link and Restart Auto-negotiation command following this command. When this field is set to 0b the device driver maintains the responsibility for sending the Setup Link and Restart Auto-negotiation command. When automatic link update is enabled, the software device driver should be aware that a link change event might occur following the Set PHY Config command.
Reserved	5.6:5.7	Reserved	Reserved, Must be zero.
EEE capability enable	6-7	EEE Capability	Sets EEE capability for each PHY type supported on the port. One bit per PHY type. The X710-TM4 might be capable of supporting multiple PHY types. The following parameter indicates the bit number. Ignores values for unsupported PHY types. Bit 6.0 = Set autonomous EEE for relevant enabled PHY types. Bit 6.1 = EEE is enabled for 100BASE-TX. Bit 6.2 = EEE is enabled for 1000BASE-T. Bit 6.3 = EEE is enabled for 10GBASE-T. Bit 6.4 = EEE is enabled for 1000BASE-KX. Bit 6.5 = Reserved. Bit 6.6 = EEE is enabled for 10GBASE-KR. Bit 6.7 = Reserved. Bit 7.0 = EEE is enabled for 2.5GBASE-T. Bit 7.1 = EEE is enabled for 5GBASE-T. Other bits = Reserved, Must be zero. This command is used by the software device driver to enable the EEE capability of various PHY types supported on the port. The EEE capability of the X710-TM4 can be read by the Get PHY Abilities command (see Section 3.2.12.1.4). The software device driver might set EEE capability for a subset of PHY types supported by the X710-TM4.
EEER	8-11	EEER Value	Value to program the EEER register.
Low Power Control	12	D3cold LPAN	D3cold LPAN. Bit 12.0 = Set to 0b to disable D3cold low power auto-negotiation. Bit 12.0 = Set to 1b to enable D3cold low power auto-negotiation. Other bits = Reserved, Must be zero.



Table 3-42. Set PHY Config Command Data Structure (Continued)

Name	Bytes.Bits	Value	Remarks
PHY Type Extension	13	PHY Type	PHY type support on port. One bit per PHY type. X710-TM4 might be capable of supporting multiple PHY types. The following parameter indicates the bit number. Ignores values for unsupported PHY types. 0 = Reserved. 1 = Reserved. 2 = Reserved. 3 = Reserved. 4 = Reserved. 5 = Reserved. 6 = 2.5GBASE-T. 7 = 5GBASE-T.
Reserved	14.0-14.4	Reserved	Reserved.
Reserved	14.5-15	Reserved	Must be set to 0x0, value is ignored.

The following structure describes the response by firmware to the [Set PHY Config](#) command.

Table 3-43. Set PHY Config Command Response (Opcode: 0x0601)

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return value. 0x0 command success. Returns EPERM code if the operation is not permitted.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Param0	16-19	Reserved	Zeroed by firmware, value is ignored.
Param1	20-23	Reserved	Must be 0x0, value is ignored.
Reserved	24-27	Reserved	Value 0x0.
Reserved	28-31	Reserved	

Note: When *Enable Automatic Link Update* is set to 1b, firmware sends a completion for the [Set PHY Config](#) command only after making all the necessary configuration changes and executing the [Setup Link and Restart Auto-negotiation](#) command.

3.2.12.1.2 Set MAC Config

This command is used by the software device driver to set the various MAC configuration parameters supported on the port. This status is indicated by the command response.

This is a direct command. The Set MAC command parameters data structure is placed in the command descriptor.



Table 3-44. Set MAC Config Command (Opcode: 0x0603)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0b, value is ignored.
Return value/VFID	6-7		Return value. Zeroed by the device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Set MAC Config	16-31	See Table 3-45	16-byte data structure that holds the Set MAC Config command parameters is listed in Table 3-45 .

[Table 3-45](#) lists the data structure of the [Set MAC Config](#) command parameters such as max frame size, etc.

Table 3-45. Set MAC Config Command Data Structure

Name	Bytes.Bits	Value	Remarks
Max Frame Size	0-1	Max Frame Size	16-bit value used to set the maximum frame size of the Ethernet frame on the port. This parameter is used by the software device driver to set the maximum frame size on the port both for Rx and for Tx. This parameter should be set to the maximum expected L2 packet size. It is ~1.5 KB or ~9.5 KB depending if jumbo packets are expected on the link. The firmware should return ERANGE error when the value provided by the software is bigger than the maximal frame size supported by the MAC (9.5 KB).
Reserved	2.0-2.1	Reserved	Must be 0x0.
CRC Enable	2.2	CRC Enable	Bit 0 = Set to 1b to enable the MAC to append the CRC on transmit. Set to 0b if software appends the CRC. This parameter is used by the software device driver to enable the MAC to append the CRC on the link. This is the default configuration. This bit is set to 0b if software needs to disable the MAC to append CRC.
Pacing Config	2.3-2.6	Pacing Config	Bit 3:0 - This is 4 bit field that allows configuring PACE parameter in the MAC to slow down the effective data rate as defined in Table 3-25 .
Reserved	2.7	Reserved	Must be 0b.
Transmit Timer Priority	3	Transmit Timer Priority	This bitmap field selects the priority <n>, with one bit per priority. The priorities selected here, are updated with the <i>Transmit Time Value</i> field and the <i>FC Refresh Threshold</i> field with the values provided in this command. For additional register description, see Section 3.2.1.5 .

**Table 3-45. Set MAC Config Command Data Structure (Continued)**

Name	Bytes.Bits	Value	Remarks
Transmit Timer Value	4-5	Transmit Timer Value	This is the priority <n> timer value that will be included in the XOFF frames being transmitted. <n> is selected in the <Transmit Timer Priority> bitmap above. <n> = 0 is used for Link Level FC. For additional register description, see Section 3.2.1.5 .
FC Refresh Threshold	6-7	FC Refresh Threshold	This field represents priority <n> FC refresh threshold, that specifies how many slot time before the XOFF expires, a new XOFF is sent. <n> is selected in the <Transmit Timer Priority> bitmap previously described. When <n> = 0, the value is used for link level flow control. This value is used to calculate the actual refresh period for sending the next pause frame if conditions for a pause state are still valid. For additional register description, see Section 3.2.1.5 .
Reserved	8-15	Reserved	Must be 0x0.

The following data structure describes the response by firmware to the [Set MAC Config](#) command.

Table 3-46. Set MAC Config Command Response (Opcode: 0x0603)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return Value. 0x0 command success. Returns EPERM code if the operation is not permitted.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31	Reserved	Value 0x0.

3.2.12.1.3 Setup Link and Restart Auto-negotiation

This command is used by the software device driver to setup the link and execute previously sent [Set PHY Config](#) commands as well as restart the auto-negotiation over the link. This command needs to be executed for any change in link parameters, such as set link speed, etc., to take effect.

This is a Direct command.

Table 3-47. Restart AN Command (Opcode: 0x0605)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return value. Zeroed by software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.

**Table 3-47. Restart AN Command (Opcode: 0x0605)**

Name	Bytes.Bits	Value	Remarks
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Command Flags	16	Command	Bit 16.1: — Set to 1 to restart the link. Bit 16.2 ¹ : — Set to 1 to enable link. — Set to 0 to disable link. Other bits = Reserved, must be zero. This command might be executed automatically by firmware, following a Set PHY Config command. In such a case these bits are assigned by firmware as follows: Bit 16.1 is set to 1. Bit 16.2 is copied from the Set PHY Config <Enable Link> field.
Reserved	17-31	Reserved	Must be 0x0, value is ignored.

- Used by the software device driver to enable/disable the link without modifying the other link settings. This is useful at POR when an application needs to have link powered down until the software device driver loads.

The following structure lists the response by firmware to the Restart Auto-negotiation command.

Table 3-48. Restart Auto-negotiation Command Response (Opcode: 0x0605)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return Value. 0x0 command success
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31	Reserved	Value 0x0.



3.2.12.1.4 Get PHY Abilities

This command is used by the software device driver to find out the various PHY abilities supported on the port.

This is an Indirect command.

Table 3-49. Get PHY Abilities Command (Opcode: 0x0600)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Length of the response buffer in bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Param0	16-19		First command parameter. 16.0 = Report qualified modules. List of qualified modules area part of the response only when this bit is set to 1b. 16.1 = Report active/init. 0b = Report the active values. For example, the values assigned to each parameter following the last Set PHY Config command. 1b = Report the initial values of the different fields. For example, the values loaded after the last reset event. All other bits are reserved.
Reserved	20-23		Must be 0x0, value is ignored.
Data Address High	24-27	Buff Addr	High bits of buffer address.
Data Address Low	28-31	Buff Addr	Low bits of buffer address.

Table 3-50 lists the get PHY abilities response structure returned by firmware to the [Get PHY Abilities](#) command.

The response, opcode and get PHY abilities response data structure buffer address are placed in the descriptor. The get PHY abilities response data structure is placed in a buffer.

**Table 3-50. Get PHY Abilities Command Response (Opcode: 0x0600)**

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return Value. EINVAL - Invalid parameters. For example, buffer too small. EIO - Error while accessing the information. EAGAIN - PHY/module interface currently busy, retry.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-19		Reserved. Must be set to 0x0.
Reserved	20-23		Reserved. Must be set to 0x0.
Data Address High	24-27	Buff Addr	Buffer Address.
Data Address Low	28-31	Buff Addr	Buffer content is listed in Table 3-48 .

[Table 3-48](#) lists the data structure of the [Get PHY Abilities](#) command response. The command response returns various PHY parameters such as PHY type, PHY ID, speed ability, pause ability, etc. The following table lists the format of the buffer content for the [Get PHY Abilities](#) reply.



Table 3-51. Get PHY Abilities Command Response Data Structure

Name	Bytes.Bits	Value	Remarks
PHY Type	0-3	PHY Type	<p>PHY type supported on the port.</p> <p>One bit per PHY type. The X710-TM4 might be capable of supporting multiple PHY types. The following parameter indicates the bit number.</p> <p>00 = SGMII 01 = Reserved. 02 = Reserved. 03 = 10GBASE-KR. 04 = Reserved. 05 = Reserved. 06 = XFI. 07 = SFI. 08 = Reserved. 09 = Reserved. 10 = Reserved. 11 = 1Reserved. 17 = 100BASE-TX. 18 = 1000BASE-T. 19 = 10GBASE-T. 20 = 10GBASE-SR (10 Gb/s SFP+ SR optical module). 21 = 10GBASE-LR (10 Gb/s SFP+ LR optical module). 22 = 10GBASE-SFP+Cu (direct attach copper). 23 = Reserved. 24 = Reserved. 25 = Reserved. 26 = Reserved. 27 = 1000BASE-SX 28 = 1000BASE-LX 29 = 1000BASE-T Optical</p> <p>Other bits - Reserved, must be zero.</p> <p>This parameter is used by the software device driver to find out the various PHY types supported on the port. The X710-TM4 might support multiple PHY types on the same port. One of the PHY types might be enabled due to the result of auto-negotiation or manually set by the firmware when auto-negotiation is disabled.</p>
Link Speed Ability	4	Link Speed	<p>Link rate supported on the port.</p> <p>Bit 4.0 = 2.5 Gb/s. Bit 4.1 = 100 Mb/s. Bit 4.2 = 1 Gb/s. Bit 4.3 = 10 Gb/s. Bit 4.4 = Reserved. Bit 4.6 = Reserved. Bit 4.7 = 5 Gb/s.</p> <p>Other bits - Reserved, must be zero.</p> <p>This parameter is used by the software device driver to find out the various link speeds supported on the port. The X710-TM4 might have the ability to support multiple link speeds on the same port. One of the link speeds might be enabled due to the result of auto-negotiation or manually set by the software device driver when auto-negotiation is disabled.</p>
Pause Ability	5.0:5.1	Pause Ability	<p>Zero returns 1b if the port supports IEEE Std. 802.3x Tx link pause or returns 0b otherwise. One returns 1b if the port supports IEEE Std. 802.3x Rx link pause or returns 0b otherwise.</p> <p>This parameter is used by the software device driver to find out the IEEE Std. 802.3x pause ability of the port.</p>
Low Power Ability	5.2	Low Power Ability	<p>1b = Low power mode. 0b= High power mode.</p>



Table 3-51. Get PHY Abilities Command Response Data Structure

Name	Bytes.Bits	Value	Remarks
Link Mode	5.3	Link Mode	0b = Link is disabled. 1b = Link is enabled.
AN Mode	5.4	AN Mode	1b = AN is enabled. 0b = AN is disabled. Report based on link mode, see Section 3.2.11.2.2.3 .
Enable Module Qualification	5.5	Enable Module Qualification	Returns 1b if a external module or PHY qualification check is enabled.
Reserved	5.6:5.7	Reserved	Reserved, must be 0b.
EEE Capability	6-7	EEE Capability	EEE capability for each PHY type supported on the port. One bit per PHY type. The X710-TM4 might be capable of supporting multiple PHY types. Bit 1 = EEE is supported for 100BASE-TX. Bit 2 = EEE is supported for 1000BASE-T. Bit 3 = EEE is supported for 10GBASE-T. Bit 4 = Reserved. Bit 5 = Reserved. Bit 6.0 = Autonomous EEE is supported. Bit 7.0 = EEE is supported for 2.5GBASE-T. Bit 7.1 = EEE is supported for 5GBASE-T. Other bits = Reserved, must be zero. This parameter is used by the software device driver to find out the EEE capability of various PHY types supported on the port. The X710-TM4 might support multiple PHY types on the same port and EEE capability is indicated for each PHY type.
EEER	8-11	EEER Value	Content of EEER register.
Low Power Control	12	D3ColdLPAN	D3cold LPAN. Bit 0 0b = D3cold low power auto-negotiation disabled. 1b = D3cold low power auto-negotiation enabled. Other bits = Reserved, must be zero.
PHY Type Extension	13	PHY Type	PHY type supported on port. One bit per PHY type. X710-TM4 might be capable of supporting multiple PHY types. The following parameter indicates the bit number. 0 = Reserved. 1 = Reserved. 2 = Reserved. 3 = Reserved. 4 = Reserved. 5 = Reserved. 6 = 2.5GBASE-T. 7 = 5GBASE-T. Note: The value of this field should come from the advertised <i>Link Speed</i> field in the NVM and X710-TM4 mode.
Reserved	14.0-14.7	Reserved	Reserved.
Current Module Extended Compliance Code	15	Extended Compliance Code	Returns the extended compliance code of the module as defined in SFP+ (Addr 0xA0, Byte 36) specification. This parameter is used by the software device driver to find out the module type on the port when connected to external modules.



Table 3-51. Get PHY Abilities Command Response Data Structure

Name	Bytes.Bits	Value	Remarks
Current PHY ID/ Vendor OUI	16-19	PHY ID/OUI	<p>This parameter is used by the software device driver to find out the PHY/module ID connected on the port.</p> <p>If the X710-TM4 is connected to an external BASE-T PHY: This four-byte field returns the {OUI, Manufacturer Model#, Revision ID} as defined in IEEE Std. 802.3-2018, 22.2.4.3.1 PHY Identifier (Registers 2 and 3). Bytes 17:16 = Register3. Bytes 19:18 = Register2.</p> <p>If the X710-TM4 is connected to an external module: This field returns the three-byte vendor OUI of the module (MSB is padded with zeros).</p>
Current Module Type	20-22	Module Type	<p>Returns the three-byte module ID.</p> <p>First byte: Module identifier. Defined by the SFP+ (Addr 0xA0, Byte 0) specification.</p> <p>Second byte: The following bits might be set to indicate the supported technologies: 0 = SFP+ Cu passive. 1 = SFP+ Cu active. 4 = 10GBASE-SR. 5 = 10GBASE-LR. 6 - 10GBASE-LRM. 7 - 10GBASE-ER. Remaining bits are reserved.</p> <p>Third byte: GbE compliance code. Defined by SFP+ (Addr 0xA0, Byte 6) specification.</p> <p>This parameter is used by the software device driver to find out the module type on the port when connected to external modules. For example, the X710-TM4 might be connected to an SFP+ or direct attached copper modules. The format of the module type returns the ID and Ethernet compliance code fields as defined in the SFP+ specification. There is no separate Ethernet compliance code for SFP+ copper modules. It is reported in a separate byte in SFP+ module. However, the X710-TM4 uses the unused bits in second byte to report SFP+ direct attach cables.</p>
Qualified Module Count	23		Number of qualified modules to be listed in the following bytes.
Qualified Module ID-n	24+n*32 - 55+n*32		<p>This is a list of qualified modules that are supported by the X710-TM4 and might be connected.</p> <p>The list contains a 24-byte field per module, based on IEEE Std 802.3 definition of device ID, containing: Vendor OUI (3 bytes). Reserved (1 bytes). Vendor Part# (16 bytes). Vendor Rev# (4 bytes). Last 8 bytes are reserved.</p>

3.2.12.1.5 Get Link Status

This command is used by the software device driver to find out the link status of the port. Firmware returns link status = up when the link is available for transmission/reception. This command also returns other operating parameters of the link such as negotiated speed, PHY type, etc.

In the X710-TM4, this is a Direct command.

**Table 3-52. Get Link Status Command (Opcode: 0x0607)**

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Command Flags	16-17	Reserved	16.1:0. 0 = NOP: LSE notification value is not modified and Get Link Status response returns the most updated value of enable/disable. 1 = Reserved. 2 = Disable link status event notification to software. 3 = Enable link status event notification to software. See Section 3.2.12.1.6 for details on LSE and enabling/disabling LSE events. All other bits are reserved. Must be 0x0, value is ignored.
Reserved	18-31	Reserved	Must be 0x0, value is ignored.

The following structure lists the response by firmware to the [Get Link Status](#) command.

Table 3-53. Get Link Status Response (Opcode: 0x0607)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Command Flags	16-17	Command Flags	Bit 0 = LSE enable: Enable link status event notification to software. Firmware sets this bit to 1b to indicate that LSE is enabled or sets to 0b if LSE is disabled. See Section 3.2.12.1.6 for further details on LSE and enabling/disabling LSE events. All other bits are Reserved. Must be 0, value is ignored
Get Link Status	18-31	See Table 3-54	14-byte data structure that holds the Get Link Status command response parameters listed in Table 3-54 .

[Table 3-54](#) lists the data structure of the [Get Link Status](#) command parameters such as link up/down, negotiated/operating speed, fault conditions, etc.



Table 3-54. Get Link Status Command Response Data Structure

Name	Bytes.Bits	Value	Remarks
PHY Type	0	PHY Type	<p>Returns operating PHY type or PHY type negotiated if auto-negotiation is enabled. The X710-TM4 might be capable of many different PHY types but only one PHY type is enabled as result of configuration or auto-negotiation. This parameter is an 8-bit integer, each value corresponds to a PHY type as follows.</p> <p>0x0 = SGMII. 0x1 = Reserved. 0x2 = Reserved. 0x3 = 10GBASE-KR. 0x4 = Reserved. 0x5 = Reserved. 0x6 = XFI. 0x7 = SFI. 0x8 = Reserved. 0x9 = Reserved. 0xA = Reserved. 0xB = Reserved. 0xC = SFP+ active optical module. 0xD = Reserved. 0xE = Unrecognized module (module doesn't match to any of the recognized modes. Link is up using the opportunistic flow.) 0xF = Unsupported module. The link is forced down. 0x11 = 100BASE-TX. 0x12 = 1000BASE-T. 0x13 = 10GBASE-T. 0x14 = 10GBASE-SR (10G SFP+ SR optical module). 0x15 = 10GBASE-LR (10G SFP+ LR optical module). 0x16 = 10GBASE-SFP+Cu (Direct attach copper). 0x17 = Reserved. 0x18 = Reserved. 0x19 = Reserved. 0x1A = Reserved. 0x1B = 1000BASE-SX. 0x1C = 1000BASE-LX. 0x1D = 1000BASE-T optical. 0x1E = Reserved. 0x1F = Reserved. 0x20 = Reserved. 0x21 = Reserved. 0x22 = Reserved. 0x23 = Reserved. 0x24 = Reserved. 0x30 = 2.5GBASE-T 0x31 = 5GBASE-T 0xFD = Unsupported Module - does not meet high temperature requirement 0xFE = Empty SFP+ cage. 0xFF = Default value - this value is returned if no PHY type has been identified yet.</p> <p>Other values are not used.</p> <p>This parameter is used by the software device driver to find out the operating PHY type on the port. One of the PHY types might be enabled due to the result of auto-negotiation/parallel detection or manually configured by firmware or software when auto-negotiation is disabled.</p>



Table 3-54. Get Link Status Command Response Data Structure

Name	Bytes.Bits	Value	Remarks
Link Speed	1	Link Speed	Returns operating link speed of the port. The PHY might be capable of many speeds but only one speed is enabled as result of configuration or auto-negotiation. This parameter is an 8-bit field, each bit corresponds to a link speed as follows. Only one bit is set at any given time. Bit 1.1 = 100 Mb/s. Bit 1.2 = 1000 Mb/s. Bit 1.3 = 10 Gb/s. Bit 1.4 = Reserved. Bit 1.5 = Reserved. Bit 1.6 = Reserved. Bit 4.0 = 2.5 Gb/s. Bit 4.7 = 5 Gb/s. Other bits = Reserved, must be zero. This parameter is used by the software device driver to find out the operating Link speed on the port. The link might be enabled at one of the link speeds due to the result of auto-negotiation/parallel detection or manually configured by firmware or software when auto-negotiation is disabled.
Link Status	2.0	Function Link Status	Returns 1b if link status = up, or returns 0b if the link status = down. This parameter indicates if the link is up and ready for data communication.
Link Fault	2.1:2.4	Link Fault	Bit 2.1 = Returns 1b if PHY has detected a link fault condition. The fault could be anywhere in the PHY layer and either on transmit or receive local or remote fault. The following bits provide additional information about a link fault condition. Bit 2.2 = Returns 1b if a transmit link fault condition is detected, 0b otherwise. Bit 2.3 = Returns 1b if a receive link fault condition is detected, 0b otherwise. Bit 2.4 = Returns 1b if a remote fault condition detected, 0b otherwise.
External Port Link Status	2.5	Port's Link Status	Returns 1b if link status = up, or returns 0b if the link status = down.
Media Available	2.6	Media Available	Returns 1b if media is available for normal link communication or returns 0b otherwise. This parameter is used by the software device driver to find out if the media is available on the port. When connected to an external module, this command returns if the media is plugged in and is available for normal communication. Note: This field is not relevant when connecting to an external 10GBASE-T PHY.
Signal Detect	2.7	Signal Detect	Returns 1b if a receive signal is detected by the PHY or module, or returns 0b otherwise. In the case of external PHYs, for example, this maps to the global signal detect function in the PHY and in some of the PHYs this maps to energy detect function on the link. In the case of external modules, this maps to the inverse of loss of signal function.
AN Completed	3.0	AN Completed	Returns 1b if auto-negotiation completed successfully or returns 0b otherwise. This bit is valid only if the PHY supports auto-negotiation and auto-negotiation is enabled.
LP AN Ability	3.1	LP AN Ability	Returns 1b if the link partner is able to perform auto-negotiation or returns 0b otherwise. This bit is valid only if the PHY supports auto-negotiation and auto-negotiation is enabled.
Parallel Detection Fault	3.2	Parallel Detection Fault	Returns 1b if the PHY detects parallel detection fault or returns 0b otherwise. This bit is valid only if the PHY supports auto-negotiation with parallel detection enabled.
FEC Enabled	3.3	FEC Enabled	Returns 1b if FEC is enabled on the link or returns 0b otherwise. This bit is valid only for backplane KR PHYs that support FEC. FEC might be enabled on the link during auto-negotiation. This bit reflects the FEC status for 10GBASE-KR FEC.



Table 3-54. Get Link Status Command Response Data Structure

Name	Bytes.Bits	Value	Remarks
Low Power State	3.4	Low Power State	0b High power mode 1b= Low power mode.
Link Pause Status	3.5:3.6	Link Pause Status	Bit 3.5 = Returns 1b if Tx link pause is enabled on the link during auto-negotiation or returns 0b otherwise. Bit 3.6 = Returns 1b if Rx link pause is enabled on the link during auto-negotiation or returns 0b otherwise. Link pause should be disabled if PFC is enabled on the link. Simultaneous operation of link pause and PFC is not supported.
Qualified Module	3.7	Qualified Module	When the X710-TM4 is connected to an external SFP+ module, this field indicates if the module is a qualified module whose OUI matches one of the pre-defined qualified modules. 0b = Module was not found in pre-configured list of qualified modules. 1b = Module is qualified.
PHY Temp Alarm	4.0	PHY Temp Alarm	Returns 1b if a temperature alarm condition is reported by the PHY or returns 0b otherwise. Typically an external PHY generates a temperature alarm condition by signaling a PHY interrupt to firmware. The temperature alarm feature should be enabled in the PHY to generate this condition.
Excessive Link Errors	4.1	Excessive Link Errors	Returns 1b if an excessive errors over the link condition is reported by the PHY or returns 0b otherwise
Port Tx Suspended	4.2-4.3	Port Tx Suspended	0 = Port's Tx active. 1 = Port's Tx suspended and drained. 2 = Reserved. 3 = Port's Tx suspended and drained. Blocked TC pipe flushed.
Reserved	4.4	Reserved	Reserved.
External PHY Download Error Code	4.5-4.7	Error Code	0x0 = No error 0x1 = Ext PHY NVM section is not present. 0x2 = Ext PHY NVM section CRC error. 0x3 = SBUS uCode load failure. 0x4 = SerDes uCode load failure. 0x5 = uCode load failure (NIMB). 0x6 = MDIO access failure. 0x7 = Initialization success.
Reserved	5	Reserved	
Loopback Level Enabled Status	5.0-5.3		0x0 = No loopback. 0x1 = MAC (IP MAC). 0x5 = BASE-T PCS. 0x6 = BASE-T External. Other values are reserved.
Loopback Type Status	5.4		0b = Local loopback (Tx->Rx) Other values are reserved.
Reserved	5.5		Reserved.
External Device Power Ability	5.6-5.7	External Device Power Class	BASE-T (bit 9.0): 0b= Low and high power. 1b= High power only.
Max Frame Size	6-7	Max Frame Size	Maximum frame size set on this port.
CRC Enable	8.2	CRC Enable	1b = CRC append is enabled on this port. 0b = CRC append is disabled on this port.
Pacing Config	8.3-8.6	Pacing Config	Bits 8.3-8.6: This is 4-bit field that enables configuring a pace parameter in the MAC to slow down the effective data rate as listed in Table 3-25 .
Reserved	8.7		Must be 0b.

**Table 3-54. Get Link Status Command Response Data Structure**

Name	Bytes.Bits	Value	Remarks
Link Type	9-12		PHY types supported on the port. This field uses the same encoding as the <i>PHY Type</i> field of the Get PHY Abilities response listed in Table 3-51 . When using a pluggable module, only PHY types supported by the current module are reported here.
Link Type Ext	13		PHY types supported on the port. This field uses the same encoding as the <i>PHY Type Ext</i> field of the Get PHY Abilities response listed in Table 3-51 . When using a pluggable module, only PHY types supported by the current module are reported here.

3.2.12.1.6 Link Status Event (Opcode: 0x0607)

The Link Status Event (LSE) is generated by firmware to the software device driver when there is a change in status in any of the event causing conditions. Event causing conditions listed in [Table 3-55](#) can be individually masked from generating LSE by using the [Set Event Mask](#) Command (See [Section 3-56](#)). The LSE uses the same command structure and link status response data structure listed in [Table 3-53](#) and [Table 3-54](#), with different command opcode. Firmware also posts this data structure to the admin receive queue with the `Flags.CMP` bit cleared indicating that this is an asynchronous event generated by firmware (such as the message is not in response to an AQ command from software).

The LSE is disabled by default, unless explicitly enabled by software. Software enables an LSE by setting the *LSE Enable* bit when issuing [Get Link Status](#) command (See [Table 3-52](#)). Firmware disables the LSE immediately after generating an LSE and does not queue further events until LSE is explicitly enabled by software by the [Get Link Status](#) command. Firmware also indicates the LSE enabled status through the *LSE Enable* bit in the [Get Link Status](#) command response data structure (See [Table 3-53](#)).

The LSE is only generated by firmware to respective PF drivers and it is software's responsibility to communicate relevant link status change events to the VF through appropriate PF to VF communication mechanisms. Software is not expected to use any hardware link status interrupt mechanisms. Hardware link status change interrupts are provided only for diagnostic use. Hence, hardware link status interrupts to PFs and VFs should be disabled for normal operation. Software should use the AQ mechanism to get the link status change notifications using the [Get Link Status](#) command and LSE.

Table 3-55. Reported Link Events

Event	Description
Link Change	Link state change. For example, the link state changes from link up to link down.
Media Not Available	Event is reported when an external module is pulled out of its cage.
Link Fault	
PHY Temperature Alarm	Event is generated when an external PHY or module generates a temperature alarm interrupt.
Excessive Errors	
Signal Detect Condition	Signal detect of link partner's laser signal indication has been asserted or de-asserted.



Table 3-55. Reported Link Events

Event	Description
Auto-Negotiation Completed	
Module Qualification Failure	When working with external modules, firmware might be enabled to perform a validation process where the module ID parameters are compared with a pre-configured, NVM loaded, list of qualified modules. If, qualification check is enabled and connected module is not found in the list, then firmware terminates the link initialization process and then generates this event.
Port Tx Suspend	Indicates that the port's Tx data path is temporarily suspended for configuration purposes.

3.2.12.1.7 Set Event Mask

This command is used by the software device driver to mask the event causing conditions of the link status event from firmware. The link status event is generated by firmware to the PF as described in [Section 3.2.12.1.6](#).

This is a Direct command.

Table 3-56. Set Event Mask Command (Opcode: 0x0613)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0, value is ignored.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-19	Reserved	Value 0x0.
Reserved	20-23	Reserved	
Event Mask	24-25	Event Mask	Masks the cause of LSE. The bit mask might be used to mask one or more event causing conditions. Set bit(s) to 1b to mask an event from causing LSE or set to 0b otherwise. The bits are cleared by default. Bit 24.0 = Reserved. Bit 24.1 = Mask link up/down condition. Bit 24.2 = Mask media not available or module not present condition. Bit 24.3 = Mask link fault condition. Bit 24.4 = Mask PHY temperature alarm condition. Bit 24.5 = Mask excessive errors over the link condition. Bit 24.6 = Mask signal detect (asserted or de-asserted) condition. Bit 24.7 = Mask auto-negotiation completed condition. Bit 25.0 = Mask module qualification failure condition. Bit 25.1 = Mask port Tx suspend. Other bits = Reserved, Must be zero.
Reserved	26-31	Reserved	Must be 0x0, value is ignored.



The following structure lists the response by firmware to the [Set Event Mask](#) command.

Table 3-57. Set Event Mask Command Response (Opcode: 0x0613)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31	Reserved	Must be 0x0, value is ignored.

3.2.12.1.8 Set Loopback Modes

This command is used by the software device driver to set various loopback modes on the port. This command is used for diagnostic or monitoring purposes only. The command should not be executed during normal operation as this might disrupt link operation.

This is a Direct command.

Table 3-58. Set Loopback Modes Command (Opcode: 0x0618)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Loopback Level	16		Loopback Level values. 0x0 = Disable loopback. 0x1 = MAC. 0x2:0x4 = Reserved 0x5 = BASE-T PCS. 0x6 = BASE-T external. Other values are reserved.
Loopback Type	17		Loopback Type Values: 0x0 = Local loopback (Tx->Rx) 0x1 = Reserved. Other values are reserved. Must be ignored when loopback level is set to disable loopback.

**Table 3-58. Set Loopback Modes Command (Opcode: 0x0618)**

Name	Bytes.Bits	Value	Remarks
Loopback Force Speed Value	18		Loopback Force Speed Value: 0x1 = 1 Gb/s. 0x2 = 10 Gb/s. 0x3 = Reserved. 0x4 = Reserved. Other values are reserved. Only applicable in MAC loopback mode and when <i>Loopback Force Speed Enable</i> is set.
Loopback Force Speed Enable	19.0		When set, the loopback speed is taken from the <i>Loopback Force Speed Value</i> . When cleared, the loopback speed is determined by the current link speed. If the link is down, or no module is present, the loopback speed is the maximum supported speed.
Reserved	19.1-31	Reserved	Must be 0x0, value is ignored.

The following structure lists the response by firmware to the Set Loopback Mode command.

Table 3-59. Set Loopback Mode Command Response (Opcode: 0x0618)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return Value. 0x0 - Command success. EPM - Operation is not permitted. EMODE - Device failed to enter the requested loopback mode. ENOSYS - Unsupported loopback configuration.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31	Reserved	Must be 0x0, value is ignored.

Note: Disabling loopback when it is already disabled results in a success response by firmware, and the loopback remains disabled.

Note: To enable one loopback mode when a different one is already active, it is recommended to first disable the current loopback mode and then enable the new loopback mode.

Note: When forcing loopback speed, the forced speed must be enabled in the link capabilities section in the NVM.

Note: When loopback is enabled, the link status is reported as Link up.

3.2.12.1.9 Set PHY Register

This command is used by the software device driver to write to an internal or external PHY register. This command should not be used in a manner that interferes with the firmware link management.

This is a Direct command.

**Table 3-60. Set PHY Register Command (Opcode: 0x0628)**

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0b, value is ignored.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Interface select	16		PHY Interface Select: 0x0 = Internal PHY. 0x1 = External PHY (MDIO interface). 0x2 = External cage (I ² C interface). Other = Reserved.
Device Address	17		The internal device address to be used in case of external PHY access. This value is ignored in case of internal PHY access.
Reserved	18.0		Reserved.
Reserved	18.1 - 19	0x0	Reserved.
Register Address	20-23		The address of the PHY register to be written.
Register Write Data	24-27		The data to write to the PHY register.
Reserved	28-31	0x0	Reserved.

The following data structure lists the response by firmware to the [Set PHY Register](#) command.

Table 3-61. Set PHY Register Response (Opcode: 0x0628)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return Value. 0x0 = Command success. Returns EPERM code if the operation is not permitted. EAGAIN - Returned if the interface is busy. EIO - Returned if the selected interface is wrong or disabled. EINVAL - Returned if invalid parameters are provided. For example, address out of range.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31	Reserved	Value 0x0.

3.2.12.1.10 Get PHY Register

This command is used by the software device driver to read an internal or external PHY register.

This is a Direct command.



Table 3-62. Get PHY Register Command (Opcode: 0x0629)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0b, value is ignored.
Return Value/ VFID	6-7		Return value. Zeroed by the device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Interface Select	16		PHY interface select: 0x0 = Internal PHY. 0x1 = External PHY (MDIO interface). 0x2 = External cage (I ² C interface). Other = Reserved.
Device Address	17		The internal device address to be used in case of external PHY access. This value is ignored in case of internal PHY access.
Reserved	18.0		Reserved.
Reserved	18.1 - 19	0x0	Reserved.
Register Address	20-23		The address of the PHY register to be read.
Reserved	24-31	0x0	Reserved.

The following data structure lists the response by firmware to the Get PHY Register command.

Table 3-63. Get PHY Register Response (Opcode: 0x0629)

Name	Bytes.Bits	Value	Remarks
Flags	1-0	0x0	See Section 7.10.5.2.1 for details.
Opcode	2-3	Opcode	Command opcode.
Datalen	4-5	0x0	Must be 0x0, value is ignored.
Return Value/ VFID	6-7		Return Value. 0x0 = Command success. Returns EPERM code if the operation is not permitted. EAGAIN - Returned if the interface is busy EIO - Returned if the selected interface is wrong or disabled EINVAL - Returned if invalid parameters are provided. For example, address out of range.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-23	Reserved	Value 0x0.
PHY Register Read Data	24-27		Data read from the PHY register.
Reserved	28-31	Reserved	Value 0x0.



3.2.13 Supported Media Types

The following table lists the supported SFP and SFP+ setups.

	Media	Description
10G	SFP+ SR / LR single-speed (10G) and multi-speed (1G / 10G) optical modules	<ul style="list-style-type: none"> These modules are identified based on the compliance code field.
	SFP+ DA twin-ax cables	
	SFP+ AoC's (Active optical Cables)	<ul style="list-style-type: none"> These modules are identified based on the compliance code field. AoC cables are enabled each time the device is enabled for 10G-SR <p>Note: Only limiting initialization cables are supported.</p>
	Breakout DA twin-ax cables or Active Copper cables	<ul style="list-style-type: none"> These modules are identified based on the compliance code field.

Table 3-64. BER and Ranges vs. Link Speed and Cable Types

Speed	Cable	Committed Reach	Committed BER
10GBASE-T	CAT-6a	Full reach: 100 m	< 10 ⁻¹²
	CAT-6a	Short reach: 30 m	
	CAT-6a	Jumper mode / direct attach: 1 m	
1000BASE-T	CAT-5e	Full reach: 100 m	< 10 ⁻¹⁰
100BASE-TX	CAT-5e	Full reach: 100 m	< 10 ⁻⁸
2.5GBASE-T	CAT-5e	Full Reach: 100 m	< 10 ⁻¹²
5GBASE-T	CAT-5e	Full Reach: 100 m	< 10 ⁻¹²

Note: Reaches listed in [Table 3-64](#) refer to real cable lengths and not to the IEEE standard model.



NOTE: *This page intentionally left blank.*



3.3 MC Interconnects

The X710-TM4 supports three sideband interfaces:

- SMBus
- Network Controller-Sideband (NC-SI)
- PCIe (together with MCTP)

The SMBus and NC-SI interfaces are described in the sections that follow. The PCIe interface is described in [Section 3.1](#).

3.3.1 SMBus

SMBus is an optional interface for pass-through and/or configuration traffic between an external MC and the X710-TM4. The SMBus channel behavior and the commands used to configure or read status from the X710-TM4 are described in [Section 9.5](#).

The X710-TM4 also enables reporting and controlling itself using the MCTP protocol over SMBus. The MCTP interface is used by the MC to only control the NIC and not for pass through traffic. All network ports are mapped to a single MCTP endpoint on SMBus. For further information, see [Section 9.7](#).

3.3.1.1 Channel Behavior

The SMBus specification defines a maximum frequency of 100 KHz. However, when acting as a slave, the X710-TM4 can receive transaction with a clock running at up to 400KHz. When acting as a master, it can toggle the clock at 100 KHz, or 400 KHz. The speed used is set by the *SMBus Connection Speed* field in the *SMBus Notification Timeout and Flags* NVM word.

3.3.2 NC-SI Interface

The NC-SI interface in the X710-TM4 is a connection to an external MC defined by the DMTF NC-SI protocol. It operates as a single interface with an external MC, where all traffic between the X710-TM4 and the MC flows through the interface.

The X710-TM4 supports the standard DMTF NC-SI protocol for both pass-through and control traffic as defined in [Section 9.6](#).

3.3.2.1 Electrical Characteristics

The X710-TM4 complies with the electrical characteristics defined in the NC-SI specification.

The X710-TM4's NC-SI behavior is configured at power-up in the following manner:

- The *Multi-Drop NC-SI* NVM bit ([Section 6.2.36.3](#)) defines the NC-SI topology (point-to-point or multi-drop; the default is point-to-point).



The X710-TM4 dynamically drives its NC-SI output signals (NC-SI_DV and NC-SI_RX) as required by the sideband protocol:

- At power-up, the X710-TM4 floats the NC-SI outputs.
- If the X710-TM4 operates in point-to-point mode, it starts driving the NC-SI outputs some time following power-up.
- If the X710-TM4 operates in a multi-drop mode, it drives the NC-SI outputs as configured by the MC.

3.3.2.2 NC-SI Transactions

The NC-SI link supports both pass-through traffic between the MC and the X710-TM4 LAN functions, as well as configuration traffic between the MC and the X710-TM4's internal units as defined in the NC-SI protocol. See [Section 9.6.2](#) for more details.



3.4 Non-volatile Memory (NVM)

3.4.1 General Overview

There are new conditions specific to the X710-TM4 that induced a new approach concerning NVM access:

- LAN traffic can be handled only if the EMP code runs.
- For flexibility reasons, the main EMP code is retrieved from the NVM and not from ROM.
- Unless authenticated, the EMP firmware code presents a potential security threat for the system:
 - Modification/tampering firmware code is presently undetectable to the rest of the system.
 - The X710-TM4's various virtualization capabilities (like SR-IOV with VFs and PFs) can reach into virtually any VM or VMM address space.
 - Malicious firmware is capable of sending out the harvested data from the server to unauthorized recipients via DMA reads ... again, undetected. The host CPU is not involved in this process.
 - Malicious firmware is capable of injecting a malware into the host, bypassing other platform-level access control.

3.4.1.1 Requirements on NVM Access

The basic requirements from NVM access in the X710-TM4 include the following:

1. Guarantee that only Intel provided firmware code (EMP) is run by the X710-TM4.
 - a. Unless the BMC overrides security version checking, Intel prevents older NVM images from replacing newer NVM images with an incremented security revision number. Note the security revision number is separate from any other version fields that might be used to distinguish versions of NVMs that modify functionality or device settings but otherwise do not present a security threat.
2. Protect NVM update flow from power failure before completion. This implies the image-update procedure of modules uses a double-bank policy.
3. Meet the boot time requirements described in [Section 4.2.1](#).
4. Guarantee that NVM settings that are critical for the X710-TM4's accessibility from the host be provided only by Intel, but can still be replaced in the field with another Intel-provided setting if necessary. This relates to the following NVM modules:
 - a. PCIe analog
 - a. RO PCIR registers auto load
 - b. PHY analog
 - c. RO PCIe LCB

Requirements 1. and 4. imply that the contents of several modules be protected via authentication while others be made RO. Refer to [Section 3.4.9](#) for details about NVM authentication.



3.4.1.2 Operational Limitations

The NVM protection method selected in the X710-TM4 relies on an authenticate on update concept. Meaning that the protected modules are not authenticated after initialization, but only before committing to a module update operation. NVM protection is guaranteed by an induction authentication chain, starting from a first secured NVM image and requiring that any step taken later on for modifying the image be secured.

This method induces several limitations and restricted working assumptions:

1. A first good EMP image is loaded into the Flash at the manufacturing site that is assumed to be safe. For example, it can be done via physical means such as nail bed.
 - a. It assumes customers (OEM and end-user) know the source of the installed components, the supply chain producing these components is not compromised during manufacturing, and after being deployed the NIC/LOM is physically protected from modification.
 - b. The possibility exists that unauthorized firmware be loaded into the X710-TM4 via physical modification post manufacturing, as well as supply chain vulnerabilities. However, firmware updates via programmatic (software) methods are enhanced to require authentication prior to updating NVM settings. Furthermore, host software can independently detect whether the firmware image has an invalid digital signature.
2. In normal operating mode, NVM write accesses are controlled by the X710-TM4 (by EMP) and cannot be performed via the memory-mapped accesses. Memory-mapped NVM access remains available for NVM read accesses only. For simplicity and flexibility reasons, NVM write accesses (except for VPD) can be initiated only via an admin command or following a BMC command, which are both handled by the EMP.
3. All supported Flash parts share the same set of opcodes used by the X710-TM4.
4. A blank Flash programming mode is provided (besides the normal programming mode previously mentioned in item 2.) where the Flash can be programmed directly without the EMP being involved via one of the legacy methods:
 - a. Memory-mapped write via host memory BAR.
 - b. Bit banging (GLNVM_FL A register).
5. The blank Flash programming mode is not safe, and must therefore be used only as a last resort to recover from initial mis-configuration. This programming mode is entered either:
 - a. When a blank Flash is detected.
 - b. When the EMP image loaded (even if authenticated by Intel) does not belong to the X710-TM4.

When entering host debug mode via a dedicated JTAG code (0x3E) or via a strapping pin combination, the host can remove address protection of the PF space and write into the register that controls the blank Flash programming mode.

3.4.2 Flash Device Requirements

The X710-TM4 merges the Niantic legacy EEPROM and Flash content in a single Flash device. Flash devices require a sector erase instruction if a cell is modified from 0b to 1b. As a result, in order to update a single byte (or block of data) it is required to erase it first. The X710-TM4 supports Flash devices with a sector erase size of 4 KB. Note that many Flash vendors are using the term sector differently. This document uses the term Flash sector for a logic section of 4 KB.

The X710-TM4 supports Flash devices that are either write-protected by default after power up or not. The X710-TM4 is responsible to remove the protection by sending the write-protection removal opcode to the Flash after power up.

The Flash part is clocked by the X710-TM4 at ~25 MHz and no fast read opcode/sequence is used.



The following opcodes are supported by the X710-TM4 as they are common to all supported Flash devices:

1. Write Enable (0x06).
2. Read Status Register (0x05).
3. Write Status Register (0x01). The written data is 0x00 to cancel the Flash default protection.
4. Read Data (0x03). Burst read is supported.
5. Byte/Page Program (0x02). To program 1 to 256 data bytes.
6. 4 KB Sector-erase (0x20).
7. Chip Erase (0xC7).
8. Read (JEDEC) Identification (0x9F).

3.4.3 Shadow RAM

NVM modules that meet one of these criteria must also be mirrored internally into a shadow RAM that is loaded once after POR:

1. Software must be able to partially update the module without being forced to rewrite the entire module (like SMBus address, VPD, etc.).

Unlike an EEPROM, Flash devices require rewriting an entire sector even if it comes to updating a single byte. The partial update is first performed against the shadow RAM. Later on, the X710-TM4 commits the entire updated shadow RAM into the Flash.

2. On device-level resets (in contrast to function-level resets), the module (or parts of it) is auto loaded by the X710-TM4 into registers that are mapped to the host memory BAR.

Auto-load done after PCIe fundamental resets (POR and PE_RST_N) must be completed within a bounded time and cannot wait for the delays involved by a sector erase operation (hundreds of ms) that could have been initiated just before. Flash read accesses are suspended until a Flash sector erase operation completes. NVM auto load performed further to device-level resets are done from the internal shadow RAM into the registers, without involving Flash read cycles.

The X710-TM4 maintains the first 32 x 4 KB sectors of the Flash for the configuration content that must be mirrored into the shadow RAM. These sectors are organized in two equally sized banks, each one capable of containing the entire shadow RAM contents. These banks are referred to as the basic NVM banks. At any time one of these two banks must be valid or else the X710-TM4 is set by hardware default and enters into blank Flash programming mode (refer to [Section 3.4.4.2](#)).

Following a Power On Reset (POR), the X710-TM4 copies the valid bank of the Flash device into the internal shadow RAM, which is made resilient to device-level resets that might occur later on. The valid bank is the lowest indexed bank with the validity field content read as 01b. The NVM *Validity* field is located at NVM Control Word 1. At any time, the valid bank is referred to as the current basic bank, while the other is referred as the next basic bank. Any further accesses of software to this section of the NVM are directed to the internal shadow RAM. Modifications made to the shadow RAM content are then copied by the X710-TM4 into the next bank of the NVM, flipping circularly the valid bank between bank 0 and bank 1 of the Flash.

This mechanism also provides a way for software to protect an image-update procedure from power down events by establishing a dual-bank policy even when performing a module partial update.

[Figure 3-3](#) shows the shadow RAM mapping and interface.

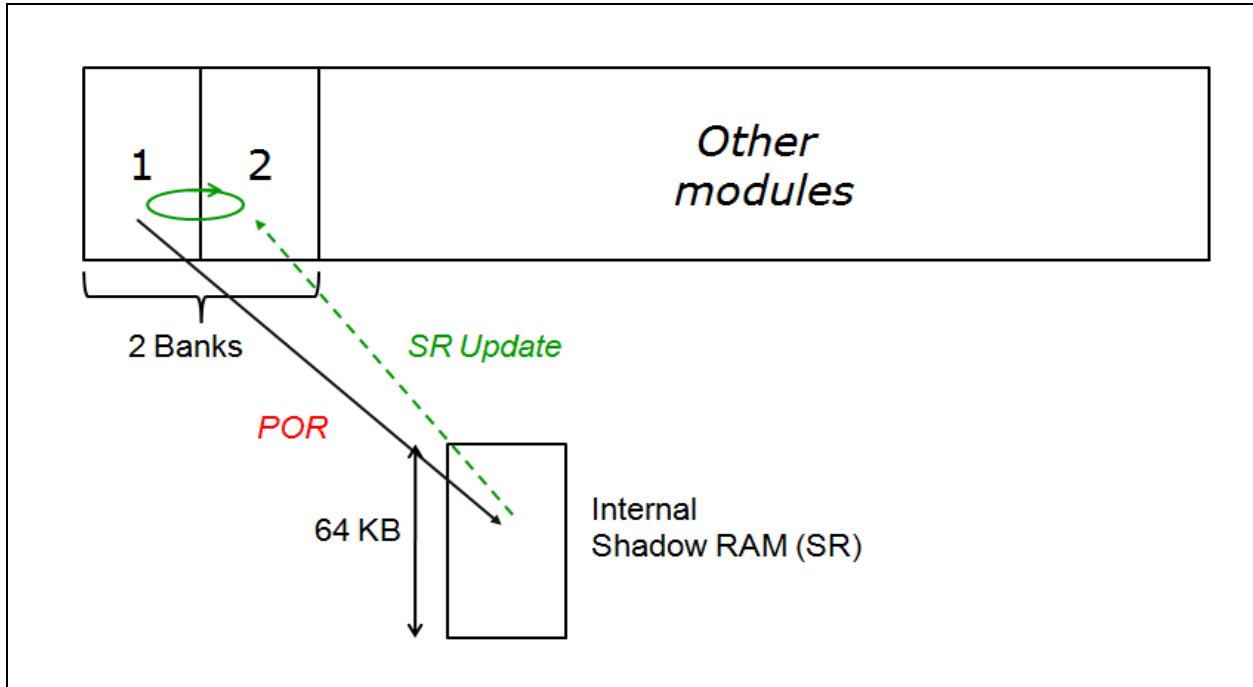


Figure 3-3. NVM Shadow RAM

3.4.4 NVM Access Modes

3.4.4.1 Normal Mode

For BIOS read accesses and VPD accesses, any read or write access to the NVM by the host must be preceded by taking ownership of the NVM resource via the Request Resource Ownership admin command. Refer to [Section 7.10.10.5](#). This prevents the following situations:

1. Reading a module that is currently being modified by another entity.
2. Concurrent modifying a module contents.

3.4.4.1.1 Normal Read Access

Read accesses to the NVM do not require the EMP being involved (except if performed via the NVM Read admin command). Available read accesses are as follows:

1. An NVM Read admin command from the PF.
2. VPD register set. The EMP asserts the *Done* bit.
3. Memory-mapped read via the memory/expansion ROM BAR. To save host memory addresses, memory BAR access to the NVM is not always available. It is enabled/disabled by setting the *Flash_Expose* bit in the NVM (or setting the `GLPCI_LBARCTRL.FLASH_EXPOSE` CSR bit).



3.4.4.1.2 Normal Write Access

Write accesses to the NVM are controlled by the EMP. Two accesses are provided:

1. An NVM Update admin command from the PF.
2. VPD register set. The EMP asserts the *Done* bit.

NVM write access attempts performed via the memory/expansion ROM BARs are not performed by the X710-TM4, although PCIe transactions are (apparently) normally completed.

3.4.4.2 Blank Flash Programming Mode

The X710-TM4 enters blank Flash programming mode based on the following:

1. When a blank Flash is detected. It means that the NVM *Validity* field (NVM Control Word 1) read from the two basic banks is not equal to 01b.
2. When the EMP image read at initialization time does not belong to the X710-TM4.
3. When host debug mode is entered via a dedicated JTAG code (0x3E) or via a strapping pin combination, the host can remove address protection of the PF space and then clear the GLNVM_FLA.LOCKED bit. It is recommended that Flash programming platforms at manufacturing sites be provided with the JTAG and/or a strapping option as a back-up means.

This mode is not safe and must be used only at manufacturing time and/or as a last resort to recover from initial mis-configurations. Only host access to the Flash and shadow RAM is guaranteed when in this mode.

It is not recommended to enter this mode at run time because no resource ownership taking is required prior to accessing the NVM. Also, taking resource ownership requires an operational EMP, which is not the case when in this mode.

When the X710-TM4 is in this mode (see steps 1. and 2. previously described), the EMP code not loaded from the NVM and the EMP remains disabled. The X710-TM4 is not able to exchange any kind of traffic over the lines and no admin command can be posted. The X710-TM4 is in an unknown operational state where only the Flash programming flow is operational.

3.4.4.2.1 Additional NVM Accesses

Additional NVM accesses are enabled while in blank Flash programming mode:

1. Direct bit-banging access to the Flash part via GLNVM_FLA register — It can be used for read and write operations of the NVM content or the Flash part Control and Status registers.
2. Memory-mapped write to the Flash via memory BAR— When in this mode, the memory BAR access to the Flash is always available since the GLPCI_LBARCTRL.FLASH_EXPOSE CSR bit is set by default.
3. The GLNVM_SRCTL, GLNVM_SRDATA register set for a write into shadow RAM.

The GLNVM_FLA.LOCKED bit — Indicates (when set to 1b) that the GLNVM_FLA register and that the memory-mapped NVM write access is locked to all software (normal programming mode). When set to 0b, software can use the bit-banging and the memory-mapped write accesses to directly access the NVM device. The bit assertion to 1b is controlled by hardware (with no EMP involved) and once asserted it is reset only on the next POR event.

3.4.5 NVM Update Flows

The flows described in this section only affect normal programming mode. When in the blank Flash programming mode, refer to NVM access procedures described in [Section 3.4.8](#).

It is assumed that an NVM update sequence does not last beyond the NVM ownership timeout for a write of three minutes. Otherwise, the sequence must be fragmented in several shorter sequences, releasing NVM ownership in between.

The software tool is responsible to re-compute and update the software checksum (word 0x3F) each time a change was made to words 0x00-0x3E further to NVM Update commands issued.

3.4.5.1 Flash High Level Map

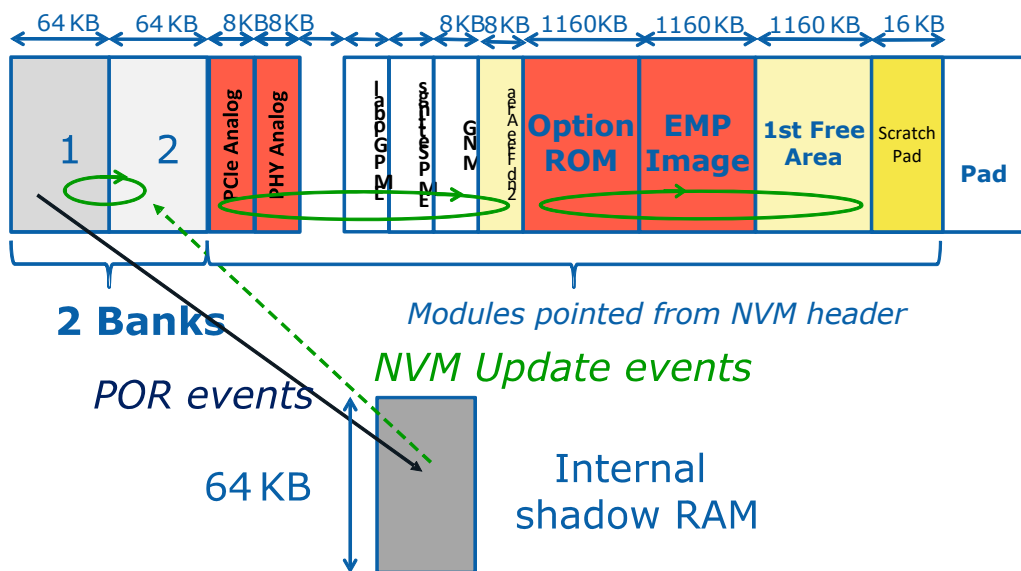


Figure 3-4. Flash High Level Map

Figure 3-4 shows the high level mapping of the NVM in the X710-TM4, while the exact mapping is given in [Chapter 6.0](#). It is made of the following areas:

- Two Basic Banks a, b — Among other modules, it contains the VPD area. The current valid bank is mirrored into the internal shadow RAM at POR events. Changes that are made in the shadow RAM are finally dumped into the next bank, which becomes the current valid, and so forth cyclically. Refer to [Section 3.4.5.2](#) and [Section 3.4.5.3](#) and [Section 3.4.5.5](#) for the shadow RAM update flows.
- EMP Image Area— It contains the firmware code to be run by the EMP embedded processor. It requires authentication before an update. Refer to [Section 3.4.5.5](#) for the update flow. The last 4 KB sector of the EMP image is covered by authentication but is not part of the compiled EMP code version. This sector contains commands to EMP for updating NVM RO words and modules. This last EMP image sector is referred as the RO commands section.
- Expansion/Option ROM Area (OROM area)— It contains pre-boot code and settings read by BIOS. Refer to [Section 3.4.5.4](#) for the update flow. Pre-boot code is authenticated by BIOS at initialization time before being executed. Pre-boot code is also internally authenticated on update.



- First Free Area — The free provisioning area used for modifying one of the 1160 KB long areas such as the EMP image area. Once the new module content has been written into the first free area, the old module area is used as the new first free area, and so forth cyclically.
- EMP Global, MNG, and EMP Settings Modules — They contain information to be handled by the EMP. Refer to [Section 3.4.5.4](#) for the update flow.
- PCIe Analog and PHY Analog Modules— These modules are only loaded by the X710-TM4 at power-up events. Refer to [Section 3.4.5.5](#) for the update flow.
- Second Free Area— The free provisioning area used for modifying one of the 8 KB long areas such as EMP settings or MNG areas. Once the new module content has been written into the second free area, the old module area is used as the new second free area, and so forth cyclically.

Refer to [Section 6.2](#) for the detailed NVM map.

3.4.5.2 VPD Update

3.4.5.2.1 First VPD Area Programming

The VPD capability is exposed on the PCIe interface only if the GLPCI_CAPCTRL.VPD_EN bit is set to 1b, regardless to any other sanity check that is performed on the VPD area contents.

The VPD area and VPD pointer must be written on a blank Flash and must contain a valid contents from this first programming. If the VPD *Write Enable* bit is set to 1b in NVM Security Control word (offset 0x02), the entire VPD area can be modified later on by a host via the NVM Update AQ command. If VPD tags were modified, it is required to issue a PCIe reset before write accessing the VPD area from PCIe configuration space.

3.4.5.2.2 VPD Area Update From PCIe Configuration Space

The flow described in this section is used once the VPD area contents and pointer have been already programmed in the NVM and loaded into the X710-TM4.

1. PF VPD software performs a VPD write — It sets write offset/data into VPD register set of the relevant PF configuration space, setting the VPD *Flag* (bit 15 in VPD Address register 0x0E2).
2. Hardware notifies the EMP — Issues an internal VPD access interrupt to the EMP to notify it of the VPD access and of the PF affected.
3. The EMP checks that the VPD write is allowed — It checks that the write offset points to the VPD-RW area (and not to a VPD RO area).
 - a. If not, the EMP clears the VPD *Flag* in the PF configuration space to notify PF VPD software that the transaction completed and then exits the flow.
4. EMP writes the change into shadow RAM
5. EMP completes the VPD access to software — The EMP clears the VPD *Flag* in the PF configuration space to notify PF VPD software that the access completed.
6. The EMP takes ownership over the NVM resources for a write — The EMP checks that the NVM is not busy and not owned by software. It then marks it internally to be owned by the EMP for a write operation, starting the 3-minute timer.
 - If the Flash is busy by a previous sector erase operation or if NVM ownership is held by software, it might be that the flow needs to be restarted from step 1. at this stage by successive VPD write accesses initiated by VPD software. That way, successive VPD writes might generate only two next bank erase operations, one for the first and one for the last VPD write in a sequence, thus increasing Flash longevity.
7. The EMP erases the next bank — It erases the contents of the next basic bank sectors, via the procedure described in [Section 3.4.8.4](#).



8. The EMP copies shadow RAM into the next bank — It copies the shadow RAM into the next basic bank sectors with the exception of the *Validity* field, which is left as all ones.
9. The EMP checks the Flash write — The new bank content is read and checked to be identical to the shadow RAM contents. This can be done in the course of writing to the Flash using previous step.
 - a. If not (such as Flash defect), exit the flow. The EMP releases the NVM ownership (internally). Refer to [Section 7.10.10.6](#).
 - b. If the check was successful, the EMP validates the new bank and invalidates the old bank.
 - The *Validity* field of the new bank is set to 01b. The EMP checks that the *Validity* field is read as written in the Flash. If not, then go to the previous sub-step.
 - The EMP toggles the state of the *BANK1VAL* bit in the *GLNVM_GENS* register to indicate that the non-valid bank became the valid one and vice versa.
 - The current (old) bank is invalidated by setting its *Validity* field to 00b.
 - The EMP releases the NVM ownership (internally). Refer to [Section 7.10.10.6](#).

Note: Users must be made aware that dumping the VPD change into the Flash might take a few hundreds of a ms after the VPD transaction completes to software (by clearing the *VPD* flag). As a result, they must wait few seconds before they can shut down the system.

If the VPD write access is attempted by the host when the X710-TM4 has just started a shadow RAM dump (step 8), then it might be that the write request times out.

3.4.5.3 Updating a RW Module or RW Words Mapped Inside Shadow RAM

Besides any RW word in the NVM header (refer to the list of RO words in [Section 6.1.1](#)), the following NVM modules are affected by this flow:

- PCIR registers auto load (pointed by NVM word 0x08).
- PBA block (pointed by NVM word 0x16).
- Boot configuration (pointed by NVM word 0x17).
- Permanent SAN MAC addresses (pointed by NVM word 0x28).
- VPD area (pointed by NVM word 0x2F) — if the VPD *Write Enable* bit is set to 1b.
- PXE setup options (pointed by NVM word 0x30).
- PXE configuration customization options (pointed by NVM word 0x31).
- VLAN configuration block (pointed by NVM word 0x37).
- POR registers auto load (pointed by NVM word 0x38).
- Software RDMA MAC address (pointed by NVM word 0x39)
- GLOBR registers auto load (pointed by NVM word 0x3B).
- CORER registers auto load (pointed by NVM word 0x3C).

The update flow is as follows:

1. Software takes ownership over the NVM resource for a write (see [Section 7.10.10.5](#)).
2. Software issues one or several NVM Update commands or NVM Config Write commands — It posts the NVM Update admin command to the EMP with the *Authenticated* bit cleared, providing the following parameters (see [Section 3.4.10.3](#)):
 - a. Address in the NVM header of the pointer to the module or 0x0000 for updating a RW word of the NVM header.
 - b. Offset inside the module.
 - c. Buffer in host memory.



- d. Buffer length (maximum supported is 4 KB).

If the *Last Command* bit is cleared in the command, it means that the command belongs to a complex NVM update operation made of several elementary NVM update commands that are posted in the admin queue. In between completions of elementary commands in a chain, other commands can be posted by a PF, besides other NVM-related commands. The entire NVM change is committed to the Flash part only once the last NVM command of the sequence is processed.

It is assumed that each command in the sequence leaves the shadow RAM with a usable/consistent contents as needed if a device-level reset occurs in the middle of the NVM update sequence.

The *Flush on Error (FE)* bit must be set for all the commands of a sequence, with the exception of the last one.

3. The EMP checks that the command is valid and posts a response to software — It performs the following command validity checks and posts a response (ACK/NACK) to software. If one check fails then the EMP flushes the remaining NVM update commands (if any) of the sequence and exits the flow. Otherwise, if no error is encountered, the EMP schedules the NVM command to run in a separate thread, resuming from the next step.
 - a. The pointer location is not a RO module or word (refer to RO words in [Section 6.1.1](#)). The size of a RO module mapped to the shadow RAM is given at the first word of the module. See [Section 6.1.5.1](#).
 - b. The start/end offsets, once applied to the module's location in the basic bank, do not lead to addresses beyond the shadow RAM size. The start/end offsets, once applied to the module's location in the basic bank, do not overwrite contents of a RO module.
 - c. If the pointer location is 0x0000, it means that the NVM update operation might affect fields in the NVM header. In that case, it must be checked that the start/end offsets does not lead to write over RO words.
4. The EMP writes the change into shadow RAM.
5. If the pointer is to the PFA (0x40), the EMP parses the modified PFA contents and performs the required configuration changes throughout the internal SR according to the meta-data section; otherwise, only the specific section within the PFA is updated.
 - a. If items loaded at EMPR only (such as MNG filter module) were modified, firmware updates them into the device RAM or registers.
6. If the *Last Command* bit is set in the command — The EMP commits the change into the next basic shadow RAM bank (a or b) by performing the following steps. Otherwise, it completes the command to software, exits the flow and goes back to step 2. when the next command is posted.
7. The EMP erases next bank — It erases the contents of the next basic shadow RAM bank sectors, via the procedure described in [Section 3.4.8.4](#).
8. The EMP copies the internal shadow RAM into next SR bank — It copies the shadow RAM into the next basic shadow RAM bank sectors with the exception of the *Validity* field, which is left as all ones.
9. The EMP checks the Flash write — The new bank content is read and checked to be identical to the shadow RAM contents. This can be done in the course of writing to the Flash using a previous step.
 - a. If not (such as Flash defect), return an error status — The command completes to software (in the next step) with the Flash defect error bit set.
 - b. If the check was successful, the EMP validates the new bank and invalidates the old bank.
 - The *Validity* field of the new bank is set to 01b. The EMP checks that the *Validity* field is read as written in the Flash. If not, then go to the previous sub-step.
 - The EMP toggles the state of the BANK1VAL bit in the GLNVM_GENS register to indicate that the non-valid bank became the valid one and vice versa.
 - The current (old) bank is invalidated by setting its *Validity* field to 00b.
10. The EMP posts an event completion on ARQ to software.
 - a. If the NVM ownership timeout for write ends before reaching this step, the EMP flushes the remaining NVM update commands (if any) of the sequence, reporting a timeout error status.
11. Software releases NVM ownership (see [Section 7.10.10.6](#)).
12. Software initiates a system reset (PE_RST_N) for loading the modifications into the X710-TM4.



In order to enable adding a new shadow RAM RW module in the future without requiring a new EMP image be loaded before, the list of concerned modules is not exhaustive, and therefore the EMP must not check that the module(s) to be updated belongs to the list.

3.4.5.4 Reprogramming a Non-authenticated Module Mapped Outside Shadow RAM

The following flow only supports the full module replacement via a double-bank policy. The following NVM modules are affected by this flow:

1. EMP global (pointed by NVM word 0x09)
2. Manageability (pointed by NVM word 0x0E)
3. PHY configuration scripts (pointed by NVM word 0x3D)
4. EMP settings (pointed by NVM word 0x0F)
5. Configuration metadata (pointed by NVM word 0x4D)
6. External PHY global (pointed by NVM word 0x4F) The contents of the modules in the list cannot alter the programming path from host to NVM, so a mis-configuration of these modules cannot permanently disable X710-TM4.

The EMP global, manageability, PHY configuration Scripts, and EMP settings modules use the second free provisioning area pointer (NVM word 0x46) for their bank swap. This free area is made up of two consecutive 4 KB sectors and it is used for the update of one module at a time. The configuration metadata and external PHY global modules use the third free provisioning area pointer (NVM word 0x44 with a size of 128 KB) for their bank swap.

1. Software takes ownership over the NVM resource for a write — Refer to [Section 7.10.10.5](#).
2. Software issues one or several NVM Erase and/or Update commands — It posts an NVM Erase/Update admin command to the EMP, providing the following parameters (see [Section 3.4.10.3](#)):
 - a. Address in the NVM of the pointer to one of the modules previously listed.

It is software's responsibility to erase a Flash sector prior to writing it.

If the *Last command* bit is cleared in the command, it means that the command belongs to a complex NVM update operation made up of several elementary NVM Update and/or Erase commands that are posted in the admin queue. In between completions of elementary commands in a chain, other commands can be posted by a PF, besides other NVM-related commands. The pointers are updated in the Flash part only once the last NVM command of the sequence is processed.

The *Flush on Error (FE)* bit must be set for all the commands of a sequence with the exception of the last one.

3. The EMP checks that the command is valid and posts a response to software — It performs the following command validity checks and posts a response (ACK/NACK) to software. If one check fails then the EMP flushes the remaining NVM update commands (if any) of the sequence and exits the flow. Otherwise, if no error is encountered, the EMP schedules the NVM command to run in a separate thread, resuming from the next step.
 - a. The pointer location belongs to one of the modules in the list.
 - b. The start/end offsets, once applied to the relevant free area, do not lead to addresses beyond the free area size.
 - c. The start/end offsets, once applied to the relevant free area, do not spread over two consecutive 4 KB sectors
4. The EMP erases/writes the free provisioning area — According to the command, the EMP erases the 4 KB sector(s) in the free provisioning area via the procedure previously described in [Section 3.4.8.4](#) or writes the change required by the (elementary) command into it.



5. If the update sequence did not end, the EMP posts an event completion on ARQ to software — If the *Last Command* bit is cleared in the (elementary) command, the EMP completes the command and then goes to back to step 3. of this flow for the processing of the next (elementary) command of the sequence.
6. If the update sequence ended, the EMP swaps pointers — If the *Last Command* bit is set in the command, the EMP swaps between the module's pointer and the free provisioning area pointers in the shadow RAM NVM header via the EMPNVM_SRCTL, EMPNVM_SRDATA MNGSB_* register set.
7. The EMP erases the next bank — It erases the contents of the next basic bank sectors, via the procedure described in [Section 3.4.8.4](#).
8. The EMP copies shadow RAM into the next bank — It copies the shadow RAM into the next basic bank sectors with the exception of the *Validity* field, which is left to all ones. It uses the internal EMPNVM_FLCTL, EMPNVM_FLDATA MNGSB_* register set.
9. The EMP checks the Flash write — New bank content is read and checked to be identical to the shadow RAM contents. This can be done in the course of writing to the Flash using a previous step.
 - a. If it is not (like Flash defect), return an error status — The EMP swaps back the two pointers in shadow RAM and the command completes to software (in the next step) with the Flash defect error bit set.
 - If the check was successful, The EMP validates the new bank and invalidates the old bank. The *Validity* field of the new bank is set to 01b. The EMP checks the *Validity* field is read as written in the Flash. If not, it goes to the previous sub-step.
 - The EMP toggles the state of the BANK1VAL bit in the GLNVM_GENS register to indicate that the non-valid bank became the valid one and vice versa.
 - The current (old) bank is in-validated by setting its *Validity* field to 00b.
10. The EMP posts an event completion on ARQ to software.
 - a. If the NVM ownership timeout for write ends before reaching this step, the EMP flushes the remaining NVM update commands (if any) of the sequence, reporting a time out error status.
11. Software releases NVM ownership (see [Section 7.10.10.6](#)).
12. Software initiates a device reset (PCIR, CORER, or GLOBR) for loading the modifications into the X710-TM4.
13. If the module modified is the manageability or the EMP settings module, then the EMP resets itself according to the flow described in [Section 4.1.2.7](#) for loading the new settings into the X710-TM4.

Note: The contents of the old module cannot be erased by a software command prior to completing this flow.

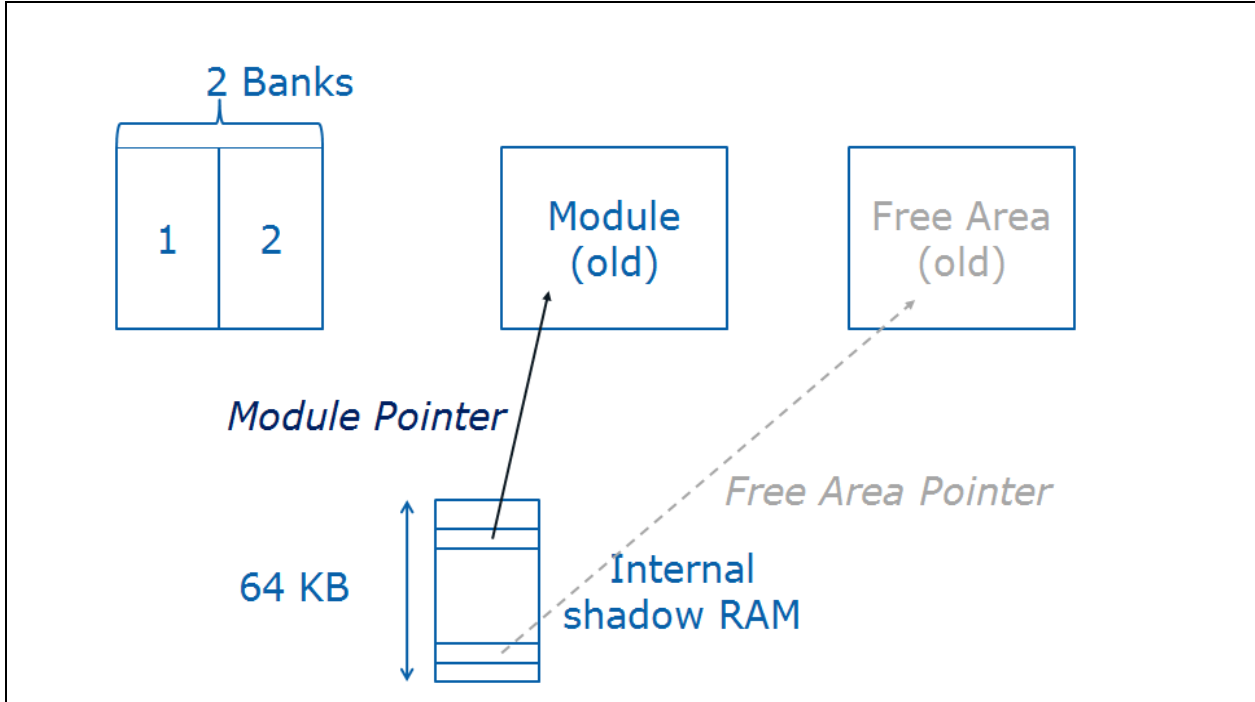


Figure 3-5. Initial State Before the NVM Update Command

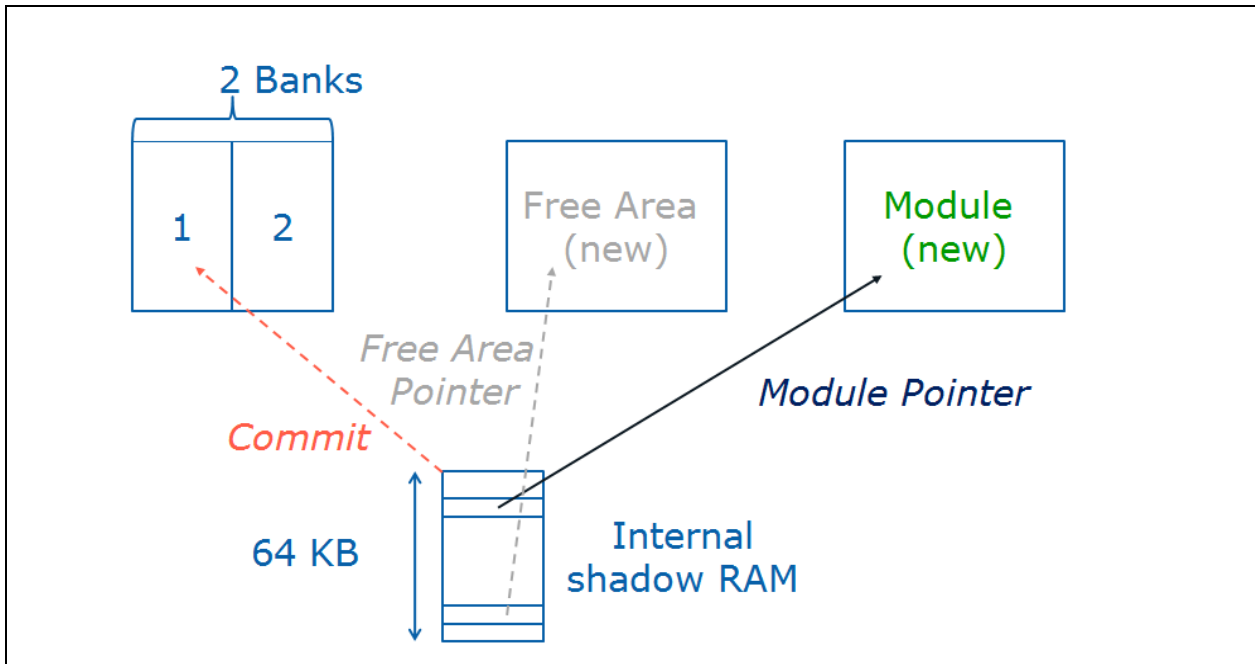


Figure 3-6. Final State After Completion of the NVM Update Command



3.4.5.5 Reprogramming an Authenticated Module Mapped Outside Shadow RAM

The following flow only supports the full module replacement via a double-bank policy. The following NVM modules are affected by this flow:

1. Option ROM (pointed by NVM word 0x05)
2. EMP image (pointed by NVM word 0x0B)
3. PCIe analog (pointed by NVM word 0x03)
4. PHY analog (pointed by NVM word 0x04)

The first free area pointer (NVM word 0x40) is used for the bank swap of the first two modules. This free area must be sized exactly to 1160 KB. The last two modules previously listed use the second free area pointer (NVM word 0x46) for their bank swap. This free area is made up of two consecutive 4 KB sectors.

1. A free area is used for the update of one single module at once. Software takes ownership over the NVM resource for a write. Refer to [Section 7.10.10.5](#).
2. Software issues one or several NVM Erase and/or Update commands — It posts the NVM Erase/Update admin command to the EMP, providing the following parameters (refer to [Section 3.4.10.3](#)):
 - a. Address in the NVM of the pointer to one of the modules previously listed.
 - b. Offset inside the module.
 - c. Buffer in host memory.
 - d. Buffer length (maximum supported is 4 KB).

It is software's responsibility to erase a Flash sector prior to writing it.

If the *Last Command* bit is cleared in the command, it means that the command belongs to a complex NVM update operation made up of several elementary NVM update and/or erase commands that are posted in the admin queue. In between completions of elementary commands in a chain, other commands can be posted by a PF, besides other NVM-related commands. The entire 1160 KB free provisioning area must always be written.

It is software's responsibility to issue the required NVM Erase commands prior to issuing NVM Update commands.

The *Flush on Error (FE)* bit must be set for all commands of a sequence with the exception of the last one.

3. The EMP checks that the command is valid and posts a response to software — It performs the following command validity checks and posts a response (ACK/NACK) to software. If one check fails then the EMP flushes the remaining NVM Update commands (if any) of the sequence and then exits the flow. Otherwise, if no error is encountered, the EMP schedules the NVM command to run in a separate thread, resuming from the next step.
 - a. The pointer location belongs to one of the modules in the list.
 - b. The start/end offsets, once applied to the relevant free area, do not lead to addresses beyond the free area size.
 - c. The start/end offsets, once applied to the relevant free area, do not spread over two consecutive 4 KB sectors.
4. The EMP erases/writes the free provisioning area — According to the command, the EMP erases the 4 KB sector in the free provisioning area via the procedure described in [Section 3.4.8.4](#) or writes the change required by the (elementary) command into it.
5. If the update sequence did not end, the EMP posts an event completion on ARQ to software — If the *Last Command* bit is cleared in the (elementary) command, the EMP completes the command and then goes back to step 3. of this flow for the processing of the next (elementary) command of the sequence.
6. The EMP checks the security revision — It checks that the security revision (*lad_srev* field in CSS header) of the new module is greater or equal to the security revision of the current module.



- a. If the check fails, the command completes to software with the *Security Revision Check Fails* bit set. The EMP then goes to step 15.
7. The EMP checks the X710-TM4 *Blank NVM Device ID* and *Module ID* fields — it checks that the the X710-TM4 *Blank NVM Device ID* field written in the new module is 0x154B. The EMP checks that the 30 LS bits of the *lad_module_id* field in the CSS header correspond to the module currently being updated.
 - a. If one of the checks fails, the command completes to software with the *EACCES* bit set. The EMP then goes to step 15.
 - b. Note that before authenticating the PCIe analog, PHY analog module, or option ROM module, the module must be appended with the last 330 words of the module's area, as the CSS header has been moved to the module's trailer. Refer to [Section 6.1.5.2](#) for more details.
8. If an update sequence ended, the EMP checks CRC8 and authenticates the module — It reads the new module from the Flash, makes sure its CRC8 is valid and then authenticates its signature according to the procedure described in [Section 3.4.9](#). This can be done in the course of writing to the Flash using the previous steps.
 - a. If one of the checks fail, the command completes to software with the relevant error bit set, either the *Public Key Check Fails* bit or the *Module Signature Check Fails* bit. The EMP then goes to step 15.
9. The EMP swaps pointers — The EMP swaps between the module's pointer and the free provisioning area pointers in the NVM header of the shadow RAM to point to the next module's bank.
10. The EMP erases next bank — It erases the contents of the next shadow RAM bank sectors, via the procedure described in [Section 3.4.8.4](#).
11. The EMP copies shadow RAM into next bank — It copies the shadow RAM into the next bank sectors with the exception of the *Validity* field, which is left as all ones.
12. The EMP checks the Flash write — The new shadow RAM bank content is read and checked to be identical to the shadow RAM contents. This can be done in the course of writing to the Flash at the previous step.
 - If not (such as Flash defect), return an error status — The EMP recovers the previous module's pointers in the shadow RAM and the command completes to software (in the next step) with the Flash defect error bit set.
 - If the check completed successfully, the EMP validates the new SR bank and invalidates the old SR bank.
 - *Validity* field of the new bank is set to 01b. The EMP checks the *Validity* field is read as written into the Flash. If not, it goes to the previous sub-step.
 - EMP toggles the state of the *BANK1VAL* bit in the *GLNVM_GENS* register to indicate that the non-valid bank became the valid one and vice versa.
 - The current (old) bank is invalidated by setting its validity field to 00b.
13. If the module updated was the EMP image,
 - a. If the last command in the flow completed with an error, go to next step to report the error.
14. The EMP posts an event completion on ARQ to software — It posts a completion/response to the last admin command of the sequence.
15. If the module version is stored in VPD, software might change the VPD module (mapped into PFA) to match the updated version loaded in the Flash.
 - a. Software tools is responsible to update the *Checksum* field's value accordingly.
16. Software releases NVM ownership. Refer to [Section 7.10.10.6](#).
17. If the module updated included the EMP image, the EMP resets itself according to the flow described in [Section 4.1.2.7](#) and reloads from the new EMP image.
 - a. Note that the new RO settings described in step 13.b. were loaded into the X710-TM4 before the new EMP code started to run.

Note: Contents of the old module cannot be erased by a software command prior to completing this flow.



3.4.6 NVM Clients and Low-level Interfaces

There are several clients that can access the NVM to different address ranges via different access modes, methods, and low-level interfaces. The various clients to the NVM are hardware, software tools (like BIOS), drivers, EMP, MC (via EMP), and VPD software.

Table 3-65 lists the different accesses to the NVM.

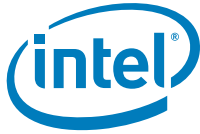
Table 3-65. Clients and Access Types to the NVM

Client	NVM Access Method	Accessed Performed Against	Logical Byte Address Range	NVM Access Interface (CSRs or Other)
VPD Software	Parallel (32-bits).	Shadow RAM	0x000000 - 0x0003FF from VPD module beginning	VPD Address and Data registers. Any write access is immediately pushed by the X710-TM4 into the Flash.
PF Software	Parallel via memory (CSR) BAR (32-bits read, 8-bits write) Write allowed only in blank Flash programming mode.	Flash Part	0x000000 - 0xFFFFFFFF	The address is relative to the beginning of the Flash. See Section 10.1.2.1 for details of the Flash offset within the CSR BAR.
	Parallel via expansion ROM BAR (32 bits read only).	Flash Part	0x020000 - 0xFFFFFFFF	This logical address range is relative to the beginning of the expansion ROM module. Write access to the Flash via expansion ROM BAR is not performed (silently dropped).
	Via AQC.	Flash Part/ Shadow RAM	0x000000 - 0x00FFFF	NVM Read, NVM Erase and NVM Update admin commands as described in Section 3.4.10 .
	Parallel (16-bits).	Shadow RAM	0x000000 - 0x00FFFF	GLNVM_SRCTL and GLNVM_SRDATA registers for a read from shadow RAM logic. Write into the shadow RAM is allowed via these registers only in blank Flash programming mode.
PF Software	Bit banging (1-bit) allowed to software, only when in blank Flash programming mode.	Flash Part	0x000000 - 0x01FFFF	GLNVM_FL_A. Accessing this range via bit-banging should be avoided during normal operation as it might cause non-coherency between the Flash and the shadow RAM.
			0x020000 - 0xFFFFFFFF	GLNVM_FL_A.

3.4.6.1 Memory-mapped Host Interface

The Flash is read (or written when in blank Flash programming mode) by the X710-TM4 each time the host CPU performs a read (or a write) operation to a memory location that is within the Flash address mapping or upon boot via accesses in the space indicated by the Expansion ROM Base Address register. Accesses to the Flash are based on a direct decode of CPU accesses to a memory window defined in either:

- Memory CSR + Flash Base Address register (PCIe Control register at offset 0x10). The Flash address space is exposed to the host memory BAR when the Flash *Expose* bit is set in the NVM (or the *GLPCI_LBARCTRL.FLASH_EXPOSE* CSR bit is set) or when the X710-TM4 is in blank Flash programming mode. The Flash size exposed is retrieved from the *GLPCI_LBARCTRL.FL_SIZE* CSR field, and is 8 MB by default (blank Flash programming mode). Refer to [Section 10.1.1.2](#) for more details.



- The Expansion ROM Base Address register (PCIe Control register at offset 0x30). The module address space is always exposed to the expansion ROM BAR unless the Flash is blank. The Flash size exposed is retrieved from the *GLPCI_LBARCTRL.EXROM_SIZE* CSR field, and is 512 KB by default. The X710-TM4 is responsible to map read accesses via the expansion ROM BAR to the physical NVM. Write attempts to the Flash through this BAR are not performed, they are silently dropped. The offset in the NVM of the expansion ROM module is defined by the PCIe expansion/option ROM pointer (Flash word address 0x05). This pointer is loaded by the X710-TM4 from the Flash before enabling any access to the expansion ROM memory space.
 - When modifying the PXE driver section pointer in the NVM, it is required to issue a PCIe reset on which the updated offset is sampled by hardware.
 - If there is no valid NVM validity field in the two basic banks, then the expansion ROM BAR is disabled.

The X710-TM4 controls accesses to the Flash when it decodes a valid access. Attempting memory-mapped write access to the Flash during normal programming mode is ignored. Out of range memory-mapped read access returns arbitrary data.

Note: Refer to [Section 3.1.2.2.1](#) for details on memory/expansion ROM BAR access rules. The X710-TM4 only supports byte writes to the Flash via the memory-mapped host interface (only when in blank Flash programming mode).

Flash read accesses are assembled by the X710-TM4 each time the access is greater than a byte-wide access.

The X710-TM4 byte reads or writes to the Flash take about 2 to 30 μ s. The X710-TM4 continues to issue retry accesses during this time.

During normal operation, the host should avoid memory-mapped accesses to the first two basic banks of the Flash because it might be non-coherent with the shadow RAM contents.

When in blank Flash programming mode, memory BAR access to the Flash while *GLNVM_FL.A.FL_REQ* is asserted (and granted) is not supported. This can lead to a PCIe hang because a bit-banging access requires several PCIe accesses.

Prior to initiating an NVM read (or write) cycle via memory mapped access, PF software is required to take ownership over the NVM resources. Refer to [Section 7.10.10.5](#).

3.4.7 Flash Access Contention

Flash read accesses initiated through PFs might occur concurrently to the EMP modifying the NVM contents. The X710-TM4 does not synchronize between the different entities accessing the Flash so contentions caused from one entity reading and the other modifying the same locations is possible.

To avoid such a contention between software and EMP accesses, these entities are required to make use of the NVM ownership taking/release flows for any read or write access to NVM. Refer to [Section 7.10.10.5](#) and to [Section 7.10.10.6](#) for more details. This is also useful to avoid the timeout of the PCIe transaction made to a memory mapped Flash address while the Flash is currently busy with a long sector erase operation.

However, two software entities cannot use the NVM ownership acquiring/release mechanisms: BIOS and VPD software.

- Since VPD software accesses only the VPD module, which is located in the first valid bank of the NVM, VPD accesses are always performed against the shadow RAM first. In this case, the EMP must take/release ownership over the NVM as if it was the originator of the Flash access. It is then hardware/EMP's responsibility to update the NVM according to the Flash update sequence described in [Section 3.4.5.2](#).



- No contention can occur between BIOS and any other software entity (VPD included) as it accesses the NVM while the operating system is down.

However, since BIOS cannot take ownership over the NVM resource, it might be that the Flash part is not accessible when BIOS attempts reading it. This might occur if a Flash erase operation was performed just before PCIe reset. In such a case, read accesses via the expansion ROM BAR returns 0xDEADBEEF.

- It is assumed that the expansion ROM signature check performed by BIOS fails in this case.
- The EMP must avoid initiating sector erase operations at boot time.
- It is assumed and recommended that users do not attempt to update the NVM contents via the BMC while the system is re-booting.
- The BMC should delay PE_RST_N de-assertion or boot running until after the BMC completed any OOB accesses to Flash memory. It is required to route the wake-up signal from the standby button to the BMC and not to the chipset. The BMC issues a system reboot signal to the chipset only after any NVM write access completes.
- If a system reboot is issued by a local user running on the host, there is no technical way to avoid contention in this case.

Note: It is the user's responsibility when accessing the NVM remotely via the BMC to make sure another user is not currently initiating a local host reboot there.

- The EMP is responsible to take NVM ownership on the BMC account prior to performing any NVM read or write access, which is needed for handling an NC-SI command. The NVM ownership is released by the EMP together with completing the NC-SI command. If NVM ownership is not free when processing the NC-SI command, the command completes with a package not ready status.

3.4.8 NVM Access Procedures

Any software read/write or EMP write flow described in this section (except flows executed by VPD software or by BIOS or to flows executed when in blank Flash programming mode) must be preceded by taking NVM ownership. Anytime software is taking NVM ownership, it must re-read the pointers to the module it plans to access because they might have been modified by the EMP in between two ownership takings.

Refer to [Section 7.10.10.5](#) and [Section 7.10.10.6](#) for the NVM ownership taking/releasing procedures as well for the associated timeouts.

3.4.8.1 Auto Load

3.4.8.2 Flash Erase Flow by the Host

This flow is available to the software PF device driver only when the X710-TM4 is in blank Flash programming mode.

1. Poll the *FL_BUSY* flag in the *GLNVM_FL A* register until cleared.
2. Set the *Flash Device Erase* bit (*FL_DER*) in the *GLNVM_FL A* register or the *Flash Sector Erase* bit (*FL_SER*) together with the Flash sector index to be erased (*FL_SADDR*).



Hardware gets the Erase command from the GLNVM_FLA register and sends the corresponding Erase command to the Flash. The erase process then finishes by itself. Software should wait for the end of the erase process before any further access to the Flash. This can be checked by polling the GLNVM_FLA.FL_BUSY bit.

3.4.8.3 Software Access to NVM via the Bit Banging Interface

This flow is available to PF software only when the X710-TM4 is in blank Flash programming mode.

To directly access the Flash, software/EMP should follow these steps:

1. Write 1b to the *Flash Request* bit (GLNVM_FLA.FL_REQ).
2. Read the *Flash Grant* bit (GLNVM_FLA.FL_GNT) until it becomes 1b. It remains 0b as long as there are other accesses to the Flash.
3. Write or read the Flash using the direct access to the 4-wire interface as defined in the GLNVM_FLA register. The exact protocol used depends on the Flash placed on the board and can be found in the appropriate datasheet.
4. Write 0b to the *Flash Request* bit (GLNVM_FLA.FL_REQ).
5. Following a write or erase instruction, software/EMP should clear the *Request* bit only after it has checked that the cycles were completed by the NVM. This can be checked by reading the *Busy* bit in the Flash device Status register. Refer to the Flash datasheet for the opcode used for reading the Status register.

Note: If software uses the bit banging interface during run time, it should adhere to the following rules:

- Gain access first to the Flash using the flow described in [Section 7.10.10.5](#).
- Minimize the GLNVM_FLA.FL_REQ setting for a single byte/word/Dword access or other method that guarantees a fast enough release of GLNVM_FLA.FL_REQ.

3.4.8.4 Flash Programming Procedure via the Memory Interface

This flow is available to PF software only when the X710-TM4 is in blank Flash programming mode.

Software initiates a write cycle via to the Flash via the memory BAR as follows:

1. Poll the *FL_BUSY* flag in the GLNVM_FLA register until cleared.
2. Write the data byte to the Flash through the memory BAR.
3. Repeat the steps 2 and 3 if multiple bytes need programming.

As a response, hardware executes the following steps for each write access:

1. Set the *FL_BUSY* bit in the GLNVM_FLA register.
2. Initiate autonomous write enable instruction.
3. Initiate the program instruction right after the enable instruction.
4. Poll the Flash status until programming completes.
5. Clear the *FL_BUSY* bit in the GLNVM_FLA register.



Note: Software must erase the sector prior to programming it. Refer to [Section 3.4.8.2](#).

Memory BAR write access to the Flash while GLNVM_FL.A.FL_REQ is asserted (and granted) is not supported. This can lead to a PCIe hang as a bit-banging access requires several PCIe accesses.

3.4.8.5 VPD Accesses

The VPD module (VPD area) is mapped into the valid basic bank and is mirrored in shadow RAM. It is accessed by VPD software via the PCIe VPD capability structure. Refer to [Section 3.4.5.2](#) for more details.

3.4.9 NVM Authentication Procedure

The NVM update integrity feature ensures that only Intel-approved firmware code (or another protected NVM module) is able to be updated on X710-TM4 devices after manufacturing. This procedure is performed by a RAM-based firmware each time an attempt is made to update one of the protected modules. Refer to the NVM update flows in [Section 3.4.5](#) for more details.

Integrity validation of NVM updates is provided by means of a digital signature. The digital signature is a SHA256 hash computed over the protected content (256 bits long) that is then encrypted by a 2048-bits RSA encryption using an Intel private key. This digital signature is stored in what is called the manifest in the NVM module image. Also stored in the manifest is the corresponding RSA modulus (public key) and RSA exponent to be used to decrypt the digital signature. Refer to [Section 6.1.5](#) for more details.

To verify the authenticity of the digital signature, EMP must first verify that the RSA *Modulus* and RSA *Exponent* fields in the new module loaded are identical to those in the current module. If the RSA *Modulus* and *Exponent* fields are the same, EMP decrypts the digital signature using the 2048-bit RSA *Modulus* and *Exponent* fields stored in the manifest of the old module to extract the expected SHA256 hash of content (stored hash). EMP then performs an independent SHA256 hash over the protected content (computed hash). If the stored hash matches the computed hash, the digital signature is accepted and the NVM module update is applied.

NVM updates are validated prior to invalidating the old NVM configuration, such that the old NVM configuration is still usable if the update fails to validate. After the new NVM is successfully verified, the updated image is committed to the X710-TM4’s Flash by the EMP.

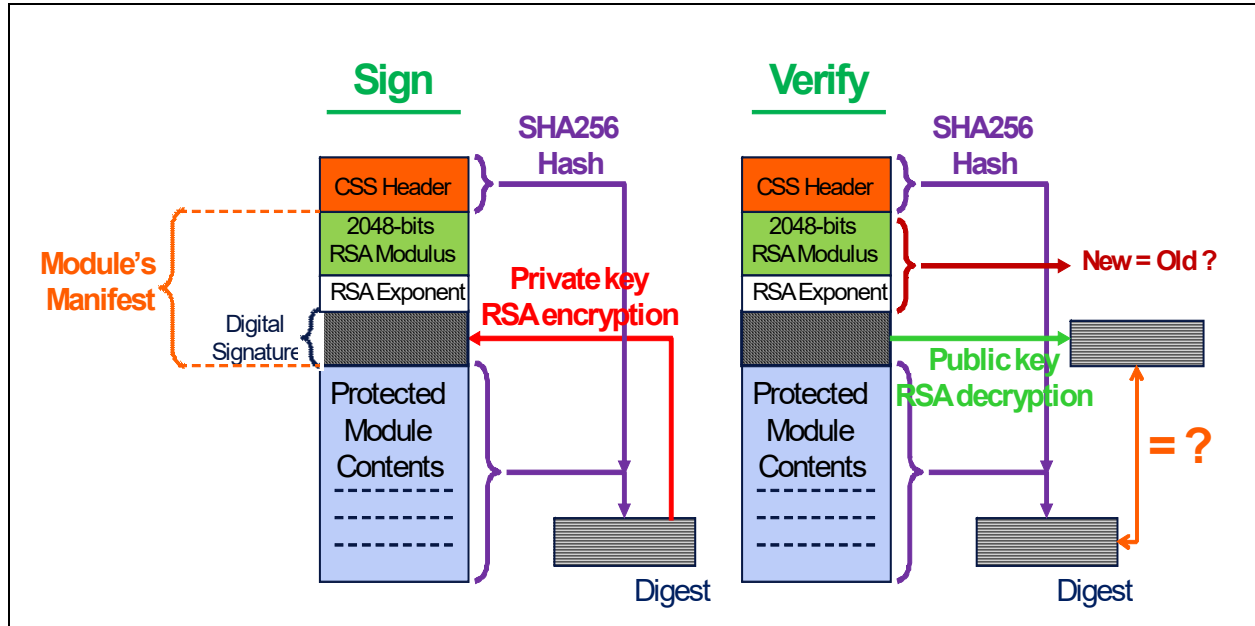


Figure 3-7. Sign and Verify Procedures for Authenticated NVM Modules

3.4.9.1 Digital Signature Algorithm Details

As previously mentioned, the digital signature generation is a hash computation followed by an RSA encryption. This is performed within Intel as part of the NVM update image generation process and not performed by Intel software in the field, nor by the X710-TM4.

The algorithms used are described in the following locations:

- PKCS #1 v2.1: RSA Cryptography Standard, RSA Laboratories, June 14, 2002 - www.rsa.com
- SHA family definition - http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf
- SHA usage with digital signatures - <http://csrc.nist.gov/publications/nistpubs/800-107/NIST-SP-800-107.pdf>
- SHA validation vectors — <http://csrc.nist.gov/groups/STM/cavp/documents/shs/SHAVS.pdf>

Note: The protected module contents shown in Figure 3-7 starts with the X710-TM4’s Blank NVM Device ID word of the NVM header described in Section 6.1.5.2 and ends with the last word of the 1160 KB long EMP image area, regardless of the size of the EMP code and to the presence and size of a RO commands section at the last sector of the EMP image area.



3.4.9.2 Intel Key Generation and Intel Code Signing System

The integrity of NVM digital signatures requires not only robust private RSA key generation but also continued protection of these private keys into the indefinite future as well as a method to generate new signed images using the existing private keys.

Intel's CSS follows the PKCSv1.5 format with 2048-bit RSA keys and SHA-256 hash. The CSS algorithm requires a standard manifest header to appear at the beginning of all signed modules. The manifest header (represented in C syntax) is as follows:

```
typedef struct _CssHeader {
    uint32 moduleType;           // Reserved CSS field
    uint32 headerLen;           // Reserved CSS field
    uint32 headerVersion;       // Reserved CSS field
    uint32 moduleID;            // Reserved CSS field.
    uint32 moduleVendor;        // Reserved CSS field
    uint32 date;                // Reserved CSS field
    uint32 size;                // Reserved CSS field
    uint32 keySize;             // Reserved CSS field
    uint32 modulusSize;         // Reserved CSS field
    uint32 exponentSize;        // Reserved CSS field
    uint32 lad_srev;            // ND-specific security revision field
    uint32 reserved;            // Reserved for future use
    uint32 lad_fw_entry_offset; // ND-specific offset from start of CSS header of the firmware
                                // code in WORDS
    int32 reserved;             // Mandatory field, in use by Springville project
    uint32 lad_image_unique_id; // ND-specific unique identifier for the specific NVM image version.
                                // Taken from NVM Software Reserved Words 22, 23 (a.k.a. EETRACK ID)
    uint32 lad_module_id;       // ND-specific and per device unique identifier for NVM module. The 30 LS
                                // bits shall be one of the following values:

    0x1 for the EMP image

    0x3 for PCIe analog

    0x5 for Option ROM

    0x6 NVM Bank

    0x7 for RO Update Module
    uint32 reserved[16];        // Mandatory field, but free for use
} CssHeader;
```

The CSS header must be placed at the beginning of the integrity-checked data. Software tools are also required to pre-process raw NVM images to prepend the manifest prior to CSS tool chain submission. All fields marked *Reserved CSS Field* must be zero when submitting to the CSS tool chain.

The X710-TM4's CSS header includes a *Security Revision (lad_srev)* field. This field is monotonically updated for each and every security-related update to the NVM. If a security exploit is detected and an updated NVM image is released with an incremental security revision, the X710-TM4 does not allow an NVM image roll back to an earlier version because this might expose previously known vulnerabilities. Security version checking can be temporarily disabled by manageability command.



Note: Not all NVM updates need to have an incremental security revision. A roll back of updates to non-security related parameters (such as firmware patches where the *Security Revision* field is equal to the existing image) are allowed.

The *Size* and *lad_nvmsize* fields give the entire module size including the CSS header itself. It is expressed in Dwords and in words, respectively. As stated in [Section 3.4.9.1](#), these fields are always set to 0x20000 and 0x40000, respectively.

3.4.9.3 Protected Modules

Any data that is modified in the field (either by the OEM during manufacturing or by the end user) cannot be included in the signed region of the NVM. The X710-TM4 cannot generate a signed image by itself because the private key is not available to it to generate the digital signature in the NVM.

Only the following NVM modules require authentication in the X710-TM4. Each module is appended by its own digital signature:

- EMP image
- PCIe analog PHY
- PHY analog
- Option ROM
- NVM bank

3.4.9.4 Software Requirements

A software tool must prepare NVM images for the CSS signing step, pre-pending the CSS manifest and applying an Intel security revision field. After receiving the signed image, the tool merges the excluded fields back into the NVM image and performs an internal integrity check to verify that the merge was successful (such as a software computation of the digital signature passes).

SVTools (LANconf) MUST implement an NVM update image integrity check option in software prior to applying NVM updates to hardware. SVTools might integrate capabilities to generate self-signed NVM images to assist in the SV and debug process by developers.

CELO (or equivalent manufacturing diagnostics tool) must implement a test to check the GLNVM_FLA *Locked* bit state as part of manufacturing qualification. If after the NVM is programmed, and the X710-TM4 powers up with the GLNVM_FLA *Locked* bit not set, an inappropriate NVM image has been loaded during manufacturing (an NVM image with incorrect Flash opcodes). This represents a critical error. With GLNVM_FLA unlocked, unauthorized in-the-field updates can bypass designed firmware integrity checks.

Host software device drivers might implement an interface enabling a network administrator to perform an internal verification check of the signed NVM image. Using Windows drivers, this would take the form of an OID, which reports a SUCCESS or INVALID_PARAMETER. Using Linux, an Ehtool command extension is advised to enable command line interrogation of the NVM content using the hash value build into the hardware as well as the saved CSS manifest in the NVM image.



3.4.9.5 Manufacturing Requirements

3.4.9.5.1 End-of-line Verification

OEM's must execute CELO (or an equivalent manufacturing diagnostics tool) to verify the GLNVM_FLG Locked bit state (as previously described).

3.4.10 NVM Access Admin Commands

NVM access commands are not supported when the X710-TM4 is in blank Flash programming mode. They are available only to the PFs once it has acquired NVM ownership via the commands described in [Section 7.10.10.5](#).

Table 3-66. NVM Access Admin Commands

Command	Opcode	Brief Description	Detailed Description
NVM Read	0x0701	Read a segment from the NVM into a host buffer	Section 3.4.10.1
NVM Erase	0x0702	Erase consecutive 4 KB sectors of the Flash	Section 3.4.10.2
NVM Update	0x0703	Write a segment of the NVM from a host buffer	Section 3.4.10.3
NVM Config Read	0x0704	Read feature selections	Section 3.4.10.4
NVM Config Write	0x0705	Program feature selections	Section 3.4.10.5

Note: All parameters in the admin commands are defined in little Endian.

3.4.10.1 NVM Read

This command is useful especially if the host memory-mapped access to the NVM was not enabled with the intent to save memory address space.

Table 3-67. NVM Read Admin Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0701	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value/ VFID	6-7		Must be zeroed by the software device driver.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.



Table 3-67. NVM Read Admin Command (Continued)

Name	Bytes.Bits	Value	Remarks
Command Flags	16		NVM Access Admin Command Parameters. Bit 0 - Last command bit, used to notify EMP that this is the last admin command of a sequence. Bits 1:6 = Reserved, must be zeroed. Bit 7 = Flash-only bit. When bit 7 is set, the read is done directly from the Flash and not from shadow RAM. Relevant only if access address is below 64 KB. Ignored otherwise.
Module_Pointer	17		Module pointer location in words from the NVM beginning. A value of 0x0000 means that the command is performed over the Flash part seen as a flat memory. In any case, the read is always performed against the Flash part and never from the shadow RAM.
Length	18-19		Length of the section to be read, which is expressed in bytes from the offset in the module. A value of 0xFFFF means the last byte to be returned is the last byte of the module (if byte 17 was not set to 0x0000). In any case, a (single) read command is limited up to 4 KB.
Offset	20-23		Byte 23 = Reserved, must be zeroed. Bytes 22:20 = Offset in the module, which is expressed in bytes from the pointed module's beginning. This is the byte offset of the first byte returned in the data buffer.
Data Address High	24-27		Address of command buffer.
Data Address Low	28-31		

Table 3-68. NVM Read Response

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0701	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value	6-7		Return Value. 0x0 = No error (success). EPMR = The module pointer location specified in the command does not permit the required operation. The word contents is not a pointer. EINVAL = Out of range offset/length (beyond the module's size). EIO = Flash defect. EBUSY = The PF is not permitted to post this command because it does not own the NVM resource. This error code is also returned if the PF attempts to post a command while another NVM command is in process.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Reserved	16		Reserved.
Module_Pointer	17		Module pointer location and copied from the command.
Length	18-19		Length to be read and copied from the command. If a value of 0xFFFF was set in the admin command (and if byte 17 was not set to 0x0000), this field returns the length from the offset to the module's end.
Offset	20-23		Byte 23 = Reserved, must be zeroed. Bytes 22:20 = Offset in the module, copied from the command.
Data Address High	24-27		Address of command buffer.
Data Address Low	28-31		



3.4.10.2 NVM Erase

This command is used to erase the contents of 4 KB Flash sectors.

Table 3-69. NVM Erase Admin Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0702	Command opcode.
Datalen	4-5		Must be zeroed by the software device driver.
Return Value/ VFID	6-7		Must be zeroed by the software device driver.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Command_Flags	16		NVM Access Admin Command Parameters. Bit 0 = Last command bit used to notify the EMP that this is the last admin command of a sequence. Bits 7:1 = Reserved, must be zeroed.
Module_Pointer	17		Module pointer location in words from the NVM beginning. A value of 0x0000 means that the command is performed over the Flash part seen as a flat memory. Attempting to erase the basic banks or RO modules area is not allowed in normal operating mode. Besides 0x0000, only the address of a free provisioning area module pointer can be listed here.
Length	18-19		Length of the section to be erased, which is expressed in 4 KB sector units from the offset in the module. The module's beginning must be aligned to a 4 KB sector for the command to be valid. A value of 0xFFFF means that the last 4 KB sector to be erased is the last sector of the free provisioning area (if byte 17 was not set to 0x0000).
Offset	20-23		Byte 23 = Reserved, must be zeroed. Bytes 22: 20 = Offset in the module, which is expressed 4 KB sector index from the pointed module's beginning.
Reserved	24-27		Reserved.
Reserved	28-31		

This command is an asynchronous command. The EMP reads the command from the ATQ and writes back an immediate completion, intended only as an ACK/NACK that the command has been addressed by the EMP. The EMP checks the validity of the command and returns an error (NACK) on the ATQ completion if it is unable to process the command. If successful (ACK), the EMP then schedules the NVM erase operation to be performed by a lower priority thread, which can be preempted by other AQ commands. Once completed, the EMP posts a completion event on the ARQ. Software must hold the NVM resource lock while performing this operation and must release it once NVM operations complete. Software must not post another NVM command while this command is in process. For example, during the time between posting the request on the ATQ and receiving the completion event on the ARQ.



Table 3-70. NVM Erase Response

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.1.2 for details.
Opcode	2-3	0x0702	Command opcode.
Datalen	4-5		Reserved.
Return Value	6-7		Return Value. 0x0 = No error (success). EPERM = The module pointer location specified in the command does not permit the required operation. The word contents is not a pointer to a free provisioning area or attempt to erase sectors from the basic banks. EINVAL = Out of range offset/length beyond the free area module's limits. ENOENT = The module pointed is not aligned with an 4 KB sector beginning. EBUSY = The PF is not permitted to post this command because it does not own the NVM resource. This error code is also returned if the PF attempts to post a command while another NVM command is in process.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Reserved	16-31		Reserved

Table 3-71. NVM Erase Completion (on ARQ)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.1.2 for details.
Opcode	2-3	0x0702	Command opcode.
Datalen	4-5		Reserved.
Return Value	6-7		Return Value: 0x0 = No error (success). EIO = Flash defect.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Reserved	16		Reserved.
Module_Pointer	17		Copied from the command.
Length	18-19		Copied from the command. If a value of 0xFFFF was set in the admin command, this field returns the length from the offset to the module's end in 4 KB units.
Offsets	20-23	0x0	Byte 23 = Reserved, must be zeroed. Bytes 22:20 = Copied from the command.
Reserved	24-27		Reserved.
Reserved	28-31		



3.4.10.3 NVM Update

This command is used to write the data given by the attached buffer into a specified location in the NVM. Erasing the relevant sector(s) by posting NVM erase command(s) (see [Section 3.4.10.2](#)) is required prior to posting this command.

Table 3-72. NVM Update Admin Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0703	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value/ VFID	6-7		Must be zeroed by the software device driver.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Command_Flags	16		NVM Access Admin Command Parameters. Bit 0 = Last command bit used to notify EMP that this is the last admin command of a sequence. Bits 6:1 = Reserved, must be zeroed. Bit 7 = Flash-only bit. When bit 7 is set, any flat write (null Module_Pointer) into the first 128 KB of the Flash is written directly to the Flash and not in shadow RAM. The bit has no effect when attempting a flat write outside the first 128 KB of the Flash.
Module_Pointer	17		Module pointer location in words from the NVM beginning. Attempting to write a RO module is not allowed. A value of 0x00 here means that the command is performed over the Flash part seen as a flat memory. No reset or pointer switch is initiated by the X710-TM4 in such a case. Any flat write attempt to the first 64 KB of the Flash is performed against the shadow RAM first, and then dumped to the next shadow RAM bank in the Flash (see Section 3.4.5.3). Any flat write attempt to the second 64 KB of the Flash is rejected as for write attempts to a RO module. A flat write attempt to the area of a RO module or to a RO word is rejected as well. A value of 0x01 in this field means that a RO update module is being provided.
Length	18-19		Length of the section to be written, which is expressed in bytes from the offset in the module. A (single) write command is limited up to 4 KB and must not spread over two (consecutive) 4 KB sectors. Also, attempting to write a RO word invalidates the entire command.
Offset	20-23		Byte 23 = Reserved, must be zeroed. Bytes 22:20 = Offset, which is expressed in bytes from the pointed module's beginning. This is the byte offset of the first byte to be written.
Data Address High	24-27		Address of command buffer.
Data Address Low	28-31		



This command is an asynchronous command. EMP reads the command from the ATQ and writes back an immediate completion, intended only as an ACK/NACK that the command has been addressed by the EMP. The EMP checks the validity of the command and returns an error (NACK) on the ATQ completion if it is unable to process the command. If successful (ACK), the EMP then schedules the NVM erase operation to be performed by a lower priority thread, which can be preempted by other AQ commands. Once completed, the EMP posts a completion event on the ARQ. Software must hold the NVM resource lock while performing this operation and must release it once NVM operations complete. Software must not post another NVM command while this command is in process. For example, during the time between posting the request on the ATQ and receiving the completion event on the ARQ.

Table 3-73. NVM Update Response

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0703	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value	6-7		Return Value. 0x0 = No error (success). EPRM = The module pointer location specified in the command does not permit the required operation. The word contents is not a pointer, or an attempt to write a RO module or word. EINVAL = Out of range offset/length (beyond the relative free area module's limits), or write spread over two (consecutive) sectors. EBUSY = The PF is not permitted to post this command because it does not own the NVM resource. This error code is also returned if the PF attempts to post a command while another NVM command is in process.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Reserved	16-31		Reserved.

Table 3-74. NVM Update Completion (on ARQ)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0703	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value	6-7		Return Value. 0x0 = No error (success). EIO = Flash defect. EACCES = Security check failed: <ul style="list-style-type: none"> • Public key check failed • Module digest check failed • Module security revision check failed • Device ID check failed • Module ID check failed
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Reserved	16		Reserved
Provisioning_ Pointer	17		Location in words from the NVM beginning with the free provisioning pointer used for the command. A value of 0x0000 is returned if the command was performed against the shadow RAM.

**Table 3-74. NVM Update Completion (on ARQ)**

Name	Bytes.Bits	Value	Remarks
Length	18-19		Copied from the command.
Offsets	20-23		Byte 23 = Reserved, must be zeroed. Bytes 22:20 = Copied from the command.
Data Address High	24-27		Address of command buffer.
Data Address Low	28-31		

3.4.10.4 NVM Config Read Admin Command

This admin command reads currently configured feature selections and immediate field values. The features/fields to be read are specified in the command's buffer. It can also be used for reading all features or fields. Feature or field iteration is used for that. The next Feature_ID / Field_ID to read are returned in the command response in this case.

Table 3-75. NVM Config Read Admin Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0704	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value/ VFID	6-7		Must be zeroed by the software device driver.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Command_Flags	16		NVM Access Admin Command Parameters. Bit 0 = single/ multiple elements: 0b = Only a single Feature_ID, Field_ID is read. 1b = Feature_ID/Field_ID iteration is used. Bit 1 = Feature/field: 0b = Feature selections are read. 1b = Immediate fields are read. Bits 2:7 = Reserved (must be set to zero).
Reserved	17	0x0	Reserved (must be set to zero)
Element count	18-19	0x0	The number of features/fields returned. Zeroed by the software device driver, written by firmware.
Feature_ID/ Field_ID	20-21	See description	Single Feature_ID/Field_ID when bit 16.0 is 0b. Feature_ID when bit 16.1 is 0b. Field_ID (LS word) when bit 16.1 is 1b. Feature_ID/Field_ID to start reading from (iterator) when bit 16.0 is 1b. Note that the Feature_ID/Field_ID = 0b is not valid and indicates that the command should read the data starting from the first Feature_ID/Field_ID in the array. This is only relevant when bit 16.0 is 1b.
Field_ID	22-23	See description	Field_ID (MS word) when bit 16.1 is 1b. Else, must be set to zero.
Data Address High	24-27		Address of command buffer.
Data Address Low	28-31		



Table 3-76. NVM Config Read Response

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0704	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value	6-7		Return Value. 0x0 = No error (success). Others - Error detected in the command.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Command_flags	16		NVM Access Admin Command Parameters. Bit 0 = Single / multiple elements: 0b = Only a single Feature_ID, Field_ID is read. 1b = Feature_ID/Field_ID iteration is used. Bit 1 = Feature/field: 0b = Feature selections are read. 1b = Immediate fields are read. Bits 2:7 = Reserved (must be set to 0x0).
Reserved	17	0x0	Reserved (must be set to zero).
Element count	18-19	See description	The number of features/fields returned.
Feature_ID/ Field_ID	20-21	See description	Single Feature_ID/Field_ID when bit 16.0 is 0b. Feature_ID when bit 16.1 is 0b. Field_ID (LS word) when bit 16.1 is 1b. Feature_ID/Field_ID to start reading from (iterator) when bit 16.0 is 1b. Note that Feature_ID/Field_ID = 0b is not valid and indicates that the command should read the data starting from the first Feature_ID/Field_ID in the array. This is only relevant when bit 16.0 is 1b.
Field_ID	22-23	See description	Field_ID (MS word). Meaningful only when bit 16.1 is 1b.
Data Address High	24-27		Address of command buffer.
Data Address Low	28-31		

3.4.10.5 NVM Config Write Admin Command

This admin command writes the feature selections and the values of the immediate fields provided in the attached command buffer to the NVM.

Table 3-77. NVM Config Write Admin Command (0x0705)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0705	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value/ VFID	6-7	0x0	Must be zeroed by the software device driver.

**Table 3-77. NVM Config Write Admin Command (0x0705)**

Name	Bytes.Bits	Value	Remarks
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Command_Flags	16		NVM Access Admin Command Parameters. Bit 0 = Reserved, zero. Bit 1 = Feature/field. 0b= Feature selections are written. 1b = Immediate fields are written. Bit 2 - Add New Configuration: 0b = Regular command. 1b = New configuration added. Bits 3:7 = Reserved, 0x0.
Reserved	17	0x0	Reserved, zero.
Element Count	18-19	See description	The number of features/fields in the command buffer.
Reserved	20-23	0x0	Reserved, 0x0.
Data Address High	24-27		Address of command buffer.
Data Address Low	28-31		

Table 3-78. NVM Config Write Response (0x0705)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0704	Command opcode.
Datalen	4-5		Length in bytes of command buffer.
Return Value	6-7		Return Value. 0x0 = No error (success). Others - Error detected in the command.
Cookie High	8-11	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the EMP into the completion of this command.
Reserved	16-31		Reserved.

3.4.10.6 NVM Config Read / Write Command Buffer

Following is the format of the command buffer attached to the NVM Config Read response and to the NVM Config Write command. The buffer can either be filled with feature selection fields or immediate field fields, depending on bit 16.1 in the command.



Table 3-79. Feature Buffer for NVM Config Read / Write

Parameter	Bytes.Bits	Description
Feature_ID	0-1	Feature_ID.
Feature Options	2-3	Feature options for NVM_Config_Read only. Reserved for NVM_Config_Write. Bit 0 = OEM only (should be set). Bit 2:1 = Reserved. Bit 3 = If set, the <i>Feature</i> fields are mapped Dword-wise, otherwise word. Bit 4 = Should be set when using a POR CSR. Bit 15:5 = Reserved.
Feature Selection/ Field Value	4-5	Configured feature selection for NVM_Config_Read. Requested feature selection for NVM_Config_Write.

Table 3-80. Immediate Buffer for NVM Config Read / Write

Parameter	Bytes.Bits	Description
Field_ID	0-3	Field_ID.
Field Options	4-5	Field options for NVM_Config_Read. Reserved for NVM_Config_Write.
Field Value	6-9	Field value.

3.4.11 VPD Support

The Flash image can contain an area for VPD. This area is managed by the OEM vendor and does not influence the behavior of hardware. Word 0x2F of the Flash image contains a pointer to the VPD area in the Flash. A value of 0b in the GLPCI_CAPCTRL.VPD_EN register bit means VPD is not supported and the VPD capability does not appear in the configuration space. The register bit must be set to 1b in the NVM only once the VPD area has been programmed. Refer to [Section 3.4.5.2.1](#).

The maximum VPD area size provisioned in shadow RAM is 1 KB but it can be smaller. The VPD block is built from a list of resources. A resource can be either large or small. The structure of these resources are listed in the following tables.

Table 3-81. Small Resource Structure

Offset	0	1 – n
Content	Tag = 0xxx,xyyyb (Type = Small(0), Item Name = xxxx, length = yy bytes)	Data

Table 3-82. Large Resource Structure

Offset	0	1 – 2	3 – n
Content	Tag = 1xxx,xxxxb (Type = Large(1), Item Name = xxxxxxxx)	Length	Data



The X710-TM4 parses the VPD structure during the auto-load process following PCIe reset in order to detect the read only and read/write area boundaries. The X710-TM4 assumes the following VPD fields with the limitations listed:

Table 3-83. VPD Structure

Tag	Length (bytes)	Data	Resource Description
0x82	Length of identifier string	Identifier	Identifier string.
0x90	Length of RO area	RO data	VPD-R list containing one or more VPD keywords.
0x91	Length of RW area	RW data	VPD-W list containing one or more VPD keywords. This part is optional.
0x78	n/a	n/a	End tag.

VPD structure limitations:

- The structure must start with a tag = 0x82.
- The structure must end with a tag = 0x78 before the shadow RAM's end. The tag must also be word aligned.
- If the X710-TM4 does not detect a value of 0x82 in the first byte of the VPD area, or if no end tag is detected, or if the structure does not follow the information listed in [Table 3-83](#), it assumes the area is not programmed:
 - Any read/write access through the VPD registers set are ignored.
 - The EMP considers the VPD area as an empty module and thus enables NVM Update commands performed over the area pointed by the VPD area pointer, up to the start of another RO module.
 - The VPD pointer itself remains RO.
- The RO area and RW area are both optional and can appear in any order. A single area is supported by per-tag type. Refer to Appendix I in the PCI 3.0 specification for details of the different tags.
- If a VPD-W tag is found, the area defined by its size is writable via the VPD structure.
- The VPD area can be accessed through the PCIe configuration space VPD capability structure listed in [Table 3-83](#). Write accesses to a RO area or any accesses outside of the VPD area via this structure are ignored. If the VPD *Write Enable* field is set to 1b in the NVM Security Control word, the entire VPD area can be modified via the NVM Update AQ command. Otherwise, the command is completed with an error status.
- VPD area must be mapped into the first valid basic bank of the Flash.
- VPD software does not check the NVM ownership before attempting to access the Flash via dedicated VPD registers (see [Section 11.3.4](#)). VPD software write access is recorded in the Flash immediately once the Flash part is available (like not busy by a previous sector erase operation). Refer to [Section 3.4.5.2](#) for more detail.



3.5 General Purpose I/O (GPIO) and LED

The X710-TM4 has a total of 30 GPIO pins that can be configured as SDPs, LED drivers or dedicated hardware functions such as for connecting to external PHYs or IEEE 1588 auxiliary devices or driving Reset on LAN (RoL). The GPIO pins can also be configured to be associated with any of the physical ports. The following sections describe the possible configurations for the GPIO pins. Many of the GPIO pins are reserved for specific use and hence named by default as SDP, LED or GPIO signals (See [Section 2.0](#)). This offers the flexibility to configure any of the GPIO pins, irrespective of their names, to different modes and associated with different ports.

The GPIO pin functionality and other attributes such as I/O direction, default value, interrupt mode etc., can be configured through Global GPIO Control registers. See the [Table 3-84](#) for the list of GPIO pins and their corresponding configuration control registers. The `GLGEN_GPIO_CTL` register has a `PIN_FUNC` field to configure the pin functionality such as SDP, LED, RoL or 1588 Timesync modes. The `PRT_NUM` field is used to configure the port number associated with the GPIO pin. The `INT_MODE` field is used to configure the GPIO pin to generate an interrupt on the rising edge, falling edge or on both edges. The `PHY_PIN_NAME` is used to indicate how the GPIO is connected to an external PHY. See the `GLGEN_GPIO_CTL` register description for additional configuration fields and options. The default value for this register is loaded from the NVM and is typically dependent on the board layout/configuration, number of ports, and PHYs present on the board.

Table 3-84. GPIO Pin Configuration Registers

GPIO Index (n= 0..29)	Pin Name	GPIO Register
GPIO0	GPIO_0	GLGEN_GPIO_CTL[0]
GPIO1	GPIO_1	GLGEN_GPIO_CTL[1]
GPIO2	GPIO_2	GLGEN_GPIO_CTL[2]
GPIO3	GPIO_3	GLGEN_GPIO_CTL[3]
GPIO4	SDP0_0	GLGEN_GPIO_CTL[4]
GPIO5	SDP0_1	GLGEN_GPIO_CTL[5]
GPIO6	SDP0_2	GLGEN_GPIO_CTL[6]
GPIO7	SDP0_3	GLGEN_GPIO_CTL[7]
GPIO8	SDP1_0	GLGEN_GPIO_CTL[8]
GPIO9	SDP1_1	GLGEN_GPIO_CTL[9]
GPIO10	SDP1_2	GLGEN_GPIO_CTL[10]
GPIO11	SDP1_3	GLGEN_GPIO_CTL[11]
GPIO12	SDP2_0	GLGEN_GPIO_CTL[12]
GPIO13	SDP2_1	GLGEN_GPIO_CTL[13]
GPIO14	SDP2_2	GLGEN_GPIO_CTL[14]
GPIO15	SDP2_3	GLGEN_GPIO_CTL[15]
GPIO16	SDP3_0	GLGEN_GPIO_CTL[16]
GPIO17	SDP3_1	GLGEN_GPIO_CTL[17]
GPIO18	SDP3_2	GLGEN_GPIO_CTL[18]
GPIO19	SDP3_3	GLGEN_GPIO_CTL[19]
GPIO20	GPIO_4	GLGEN_GPIO_CTL[20]

**Table 3-84. GPIO Pin Configuration Registers (Continued)**

GPIO Index (n= 0..29)	Pin Name	GPIO Register
GPIO21	GPIO_5	GLGEN_GPIO_CTL[21]
GPIO22	LED0_0	GLGEN_GPIO_CTL[22]
GPIO23	LED0_1	GLGEN_GPIO_CTL[23]
GPIO24	LED1_0	GLGEN_GPIO_CTL[24]
GPIO25	LED1_1	GLGEN_GPIO_CTL[25]
GPIO26	LED2_0	GLGEN_GPIO_CTL[26]
GPIO27	LED2_1	GLGEN_GPIO_CTL[27]
GPIO28	LED3_0	GLGEN_GPIO_CTL[28]
GPIO29	LED3_1	GLGEN_GPIO_CTL[29]

Note: GPIO22 - GPIO29 are used for LED functionality only. As a result, the GLGEN_GPIO_CTL[29:22].PIN_FUNC field must be set to LED (001b).

The following registers are used to read and write to the GPIO pins. The GLGEN_GPIO_STAT register is used to read the status of the GPIO pins. This register returns the actual value (0b = high, 1b = low) on the pin. The GLGEN_GPIO_TRANSIT register is used to latch any transition (low-to-high or high-to-low) on the GPIO pins since the last time the register was cleared. The GLGEN_GPIO_SET register is used to set the value (0b = low, 1b = high) of the GPIO output pins when configured as an SDP. The PF (software) is expected to write to a GPIO pin only if it owns it. See [Section 7.10.10.5](#) and [Section 7.10.10.6](#) for details on requesting ownership of shared resources. When any of the GPIO pins are configured to generate interrupts, the PFINT_GPIO_EN and EMPINT_GPIO_EN registers are used to enable or disable the interrupts to the PF or EMP that owns the pins.

3.5.1 LEDs

The X710-TM4 designates eight pins named LED0_0 through LED3_1 for driving LEDs for ports 0 through 3 as described in [Section 2.2.6.1](#). However, depending on the board layout, these might be replaced or complemented with any of the GPIO pins by configuring the GPIO to be a LED driver through the GLGEN_GPIO_CTL register as described in [Section 3.5.3](#).

In order to configure a GPIO pin as an LED driver using the GLGEN_GPIO_CTL register:

- Set PIN_FUNC to LED
- Set PIN_DIR to output
- Use the PORT_NUM field to assign a port number to the LED pin

The output polarity of the LED pin (active high or active low) can be configured through the *LED_INVRT* field and the blink mode (blinking versus steady) is configured through the *LED_BLINK* field. The LED outputs can be individually configured to indicate a particular event, state, or activity by using the *LED_MODE* field. See [Table 3-85](#) for more information on configuring the LED source mode field. The hardware default configuration for GPIO pins configured as LED outputs can be specified via NVM fields thereby supporting LED displays configurable to a particular OEM preference. Note that at run time there is usually no need to override the LED setting, the only exception would be to force a LED to blink in order to physically identify a port.



Table 3-85. LED Source Modes

LED_MODE Value	Mode	Condition ¹
LED_MODE[4] = 0		
0000b	LED_OFF	Always off.
0001b	LINK	True while link is held.
0010b	Reserved	Reserved.
0011b	LINK_10G	True while the X710-TM4 has 10 Gb/s link.
0100b	LINK_1G	True while the X710-TM4 has 1 Gb/s link.
0101b	LINK_100	True while the X710-TM4 has 100 Mb/s link.
0110b	LINK_10G	True while the X710-TM4 has 10 Gb/s link.
0111b	LINK_1G	True while the X710-TM4 has 1 Gb/s link.
1000b	LINK_10G/1G	True while the X710-TM4 has 10 Gb/s or 1 Gb/s link.
1001b	LINK_10G	True while the X710-TM4 has 10 Gb/s link.
1010b	Combined Ports Activity	Active when any one of the X710-TM4's ports has an established link with packets being transmitted or received. In this mode, LED_BLINK must be set.
1011b	Reserved	Reserved.
1100b	LINK_ACT	Asserted steady when link is established and there is no transmit or receive activity. Blinking when there is link and receive or transmit activity. In this mode LED_BLINK must be cleared at 0b.
1101b	MAC_ACT	Active when link is established and packets are being transmitted or received. In this mode, the LED_BLINK must be set.
1110b	FILTER_ACT	Active when link is established and packets are being transmitted or received that passed MAC filtering. In this mode, the LED_BLINK must be set.
1111b	LED_ON	Always true. Overrides all other settings.
LED_MODE[4] = 1 (firmware LED)		
Note: The following LED modes do not support LED_BLINK and the configuration is ignored when a LED is configured as one of the following:		
0000b	AGGREGATE_LINK_10G	Asserted when all enabled ports have 10 Gb/s link.
0001b	COMBINED_LINK	Asserted when any of the enabled ports has link.
0010b	LINK_MAX_SPEED	On when Base-T link speed is at the maximum that is available on this port. Off otherwise.
0011b	LINK_NOT_MAX_SPEED	On when Base-T link speed is not at the maximum that is available on this port. Off otherwise.
0100b	LINK_NOT_MAX_SPEED_100M_OFF	On when the link speed is not at the maximum that is available on this port and not 100 Mb/s. Off when the link speed on this port is 100 Mb/s or the link is down.
0101b	COMBINED_LINK_10G_NOT_ALL	Asserted when at least one of the enabled ports has 10 Gb/s link but not all. Note: This LED is never asserted when only one port is enabled.
0110b	LINK_MAX_SPEED	On = When link speed is the maximum that is available on this port. Off = Otherwise.
0111b	LINK_NOT_MAX_SPEED	On = When link speed is not the maximum that is available on this port. Off = Otherwise.
1000b	LINK_NOT_MAX_SPEED_100M_OFF	On = When link speed is not the maximum that is available on this port and not 100 Mb/s. Off = When link speed on this port is 100 Mb/s or the link is down.

1. When the condition is true, the LED might blink or be constantly on, depending on the value of LED_BLINK field.



The `LED_INVRT` bit enables the LED source to be inverted before being output or observed by the blink-control logic. LED outputs are assumed to normally be connected to the negative side (cathode) of an external LED.

The `LED_BLINK` bit controls whether the LED should blink while the LED source is asserted. The blink control can be especially useful for ensuring that certain events, such as `ACTIVITY` indication, cause LED transitions, which are sufficiently visible to the human eye. The global blink mode bit in the `GLGEN_LED_CTL` register is used to select between blinking every 200 ms or every 83 ms.

Note: The `LINK/ACTIVITY` source functions slightly different from the others when `BLINK` is enabled. The LED is:

- Off if there is no `LINK`
- On if there is `LINK` and no `ACTIVITY`
- Blinks if there is `LINK` and `ACTIVITY`

Note: One software device driver should override any of the LED settings at any time (host-wide). Since this is an administrator specified override, hardware does not enforce this. Typically, there is no need to override the LED hardware defaults loaded from the NVM in a specific board configuration.

3.5.2 Software-Definable Pins (SDPs)

The X710-TM4 enables any of the GPIO pins to be configured as SDPs through the `GLGEN_GPIO_CTL` register as described in [Section 3.5.3](#). The X710-TM4 has designated 16 GPIO pins, `SDP0_0` (GPIO4) through `SDP3_3` (GPIO19), as SDPs for use with ports 0 through 3, as described in [Section 2.2.6.2](#). However, depending on the board layout, any of the GPIO pins, irrespective of their name, could be configured as SDPs. In order to configure a GPIO pin for SDP use, set `PIN_FUNC` to `SDP` and `PIN_DIR` to either input or output in the `GLGEN_GPIO_CTL` register. Use the `PORT_NUM` field to assign a port number to be associated with the SDP.

The X710-TM4 SDP pins can be used for hardware connectivity to low-speed, optical-module interfaces, external PHY control or software controllable purposes. The SDP pins can also be configured for use as external interrupt sources. The SDP pins, port ownership, direction and their functions are bound to specific board layout/configuration. The default values for GPIO control registers (`GLGEN_GPIO_CTL[n]`) are loaded from the NVM. Typically, there is no need for software to change the default configuration assigned for a specific board layout. Pins assigned for PHY management are owned by firmware and should not be accessed directly by software.

The X710-TM4 SDP pins can also be configured for use as general purpose external interrupt sources (GPI). To act as GPI pins, the desired pins must be configured as inputs and enabled by the `INT_MODE` field in the `GLGEN_GPIO_CTL` register. The `INT_MODE` field can be used to configure the pin to generate an interrupt on the rising edge, falling edge or on both edges. Rising or falling edge detection occurs by comparing values sampled at the internal clock rate, as opposed to an edge-detection circuit. When detected, a GPIO interrupt is indicated in the `PFINT_ITR0` register. The `PFINT_GPIO_ENA` register is used to enable interrupts to PFs (software) and the `EMPINT_GPIO_ENA` register is used to enable an interrupt to the EMP (firmware). `GLGEN_GPIO_STAT` and/or `GLGEN_GPIO_TRANSIT` registers can be used to read the status of the GPIO pins that caused the interrupt. Software is expected to enable interrupts only for pins that are owned by the PFs.

Software or firmware can read/write to a GPIO/SDP pin as described in [Section 3.5.3](#).

The proposed use of SDPs for controlling external PHY devices or modules is explained in [Section 3.2.10.5](#). Proposed use of the SDPs for Package ID setting is explained in [Section 9.2.2.3.1](#). Proposed use of the SDPs for 1588 functionality is explained in [Section 8.5.5.1](#) and other use cases.



3.5.3 Global GPIO Pins

The X710-TM4 has designated six general purpose I/O pins as Global GPIO pins named GPIO_0 through GPIO_5, as listed in [Section 2.2.6.3](#). These pins are not assigned for a specific usage or port. The mode and port association for these pins can be configured as described in [Section 3.5](#). These pins can be used as SDP, LED, or 1588 Timesync modes or as general purpose interrupt sources. System designers have the flexibility to assign these GPIO pins for specific use depending on the board configuration.



4.0 Initialization

4.1 Reset Operation

The following sections lists the hardware and software reset sources that initialize the entire portions of the X710-TM4 and functions level resets. The reset sources are listed in Section 4.1.1 and the reset flows are detailed in Section 4.1.2.

4.1.1 Reset Sources

This section lists the reset sources supported by the X710-TM4 while the complete list of initialized logic is listed in Table 4-1. Hierarchical reset tree is shown in the Figure 4-1. Any logic initialized by a specific reset is also initialized by any other reset source that is linked to it and located above it in the reset tree.

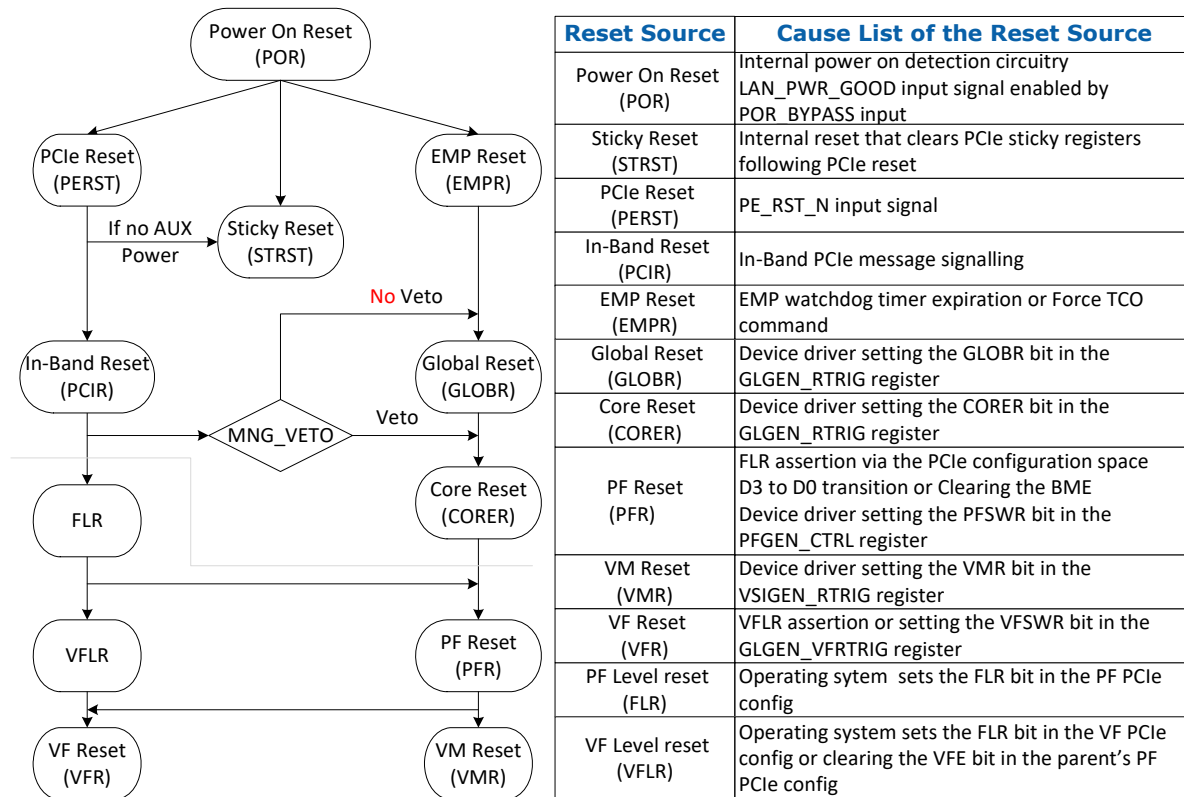


Figure 4-1. Hierarchical Reset Tree



Power On Reset (POR) — The X710-TM4 has an internal POR signal that transitions from low to high when the device power sources are above 85% of the nominal operation voltage and the internal clocks are stable. As long as the internal POR signal is at the low level, the entire device is held at the reset state. On a transition of the POR to high level, the X710-TM4 starts an internal initialization sequence that impacts the entire device. The device also has a LAN_PWR_GOOD input signal that might be used instead of the internal POR depending on the POR_BYPASS input signal setting. If the POR_BYPASS is set to 0b the internal POR is used. Otherwise, the external LAN_PWR_GOOD signal is used.

PCIe Reset (PE_RST_N) — The PCIe reset signal is kept at the low level when the system is at power down state and at system boot. Transitioning PCIe reset to low generates an internal reset signals. If there is no AUX power, PE_RST_N generates an internal STRST signal that clears PCIe sticky bits as listed in [Table 4-1](#). PE_RST_N also internally triggers a PCIR, which is detailed later in this section.

Sticky Reset (STRST) — Sticky reset is internal signal triggered by PCIe reset when the X710-TM4 is not powered by AUX power. This reset clears sticky registers in the PCIe interface (as defined by the PCIe specification). The sticky reset is also initiated by POR.

In-Band Reset (PCIR) — The PCIe supports in-band signaling for PCIe reset (called PCIR). Any cycles on the PCIe bus are gated instantly as well and packet transmission generated by software. The entire data path is initialized other than the EMP cluster. Although the EMP subsystem is not initialized, some packets of the EMP might be lost during a short window of about 1 μ s.

Function Level Reset (FLR) — The X710-TM4 supports the standard FLR interface in the Device Control register of the PCIe capability structure within the PCI configuration space of the PFs. Setting the *FLR* bit initializes the PCI configuration of the PF (including the VFE flag in the SR-IOV Control/Status register that disables all the VFs of the PF) and initiates an internal PFR described later in this section.

PF Reset (PFR) — PFR initializes the resources and data path of the PF and its VFs with no impact on other PFs, VFs and the EMP subsystem. Any further master cycles of the PF are not initiated while some packets that were already fetched completely might still be sent out. The PFR is generated by one of the following four causes:

1. D0 to D3hot transition, which is also known as ACPI reset.
2. FLR.
3. PF software sets the *PFSWR* bit in the PFGEN_CTRL register.

VF Level Reset (VFLR) — The X710-TM4 supports the standard VFLR interface in the Device Control register of the PCIe capability structure within the PCI configuration space of the VFs. Setting the *VFLR* bit initializes the PCI configuration of the VF and initiates an interrupt to the PF that completes the VFR reset described later in this section. Clearing the *VFE* flag in the PF configuration space also impacts all its VFs the same as a VFLR.

VF Reset (VFR) — VFR initializes the resources and data path of the VF with no impact on other PFs, VFs and the EMP subsystem. Any further master cycles of the VF are not initiated while some packets that were already fetched completely might still be sent out. The VFR is generated by one of the following two causes:

1. VFLR or clearing the *VFE* bit of the parent PF. Note that after the *VFE* bit clears, the PF driver should follow the VFR flow for all the VFs of the PF, including those VFs that are not enabled.
2. PF software sets the *VFSWR* bit in the VPGEN_VFRTRIG register of its VF.

VM Reset (VMR) — There are 384 VSIs where up to 256 of them can be VMDq2 VSIs (see [Section 7.4.5.2.1.1](#)). Such VSIs are associated with VMs. VMR is a mechanism to reset a VMDq2 VSI. VMR initializes the resources and data path of the VM with no impact on other PFs, VFs, VMs and the EMP subsystem. Any further master cycles of the VM are not initiated while some packets that were already fetched completely might still be sent out. The VMR is generated by the PF software setting the *VMSWR* bit in the VSIGEN_RTRIG register.



Core Reset (CORER) – CORER initializes the shared data path for all functions excluding the EMP subsystem, PCI interface and MAC/PHY logic of all ports. Any further master cycles of all PFs and VFs are not initiated while some packets that were already fetched completely might still be sent out. Even though the EMP subsystem is not cleared, pass-through traffic might be inhibited during the initialization cycle that might take ~20 ms. Also, SMBus accesses are responded with NACK during the initialization cycle. This reset is not expected to be used other than as an escape mechanism in case the X710-TM4 hangs and the PFR did not resolve the problem. This reset is initiated by the PFs by setting the *CORER* bit in the *GLGEN_RTRIG* register. The EMP initiates this reset by setting a bit in an internal register.

Global Reset (GLOBR) – GLOBR is a super-set CORER initializing any logic initialized by the CORER plus the MAC/PHY logic of all ports (both internal PHY and external PHY if connected). This reset is not expected to be used other than escape mechanism in case CORER did not resolve the problem. This reset is initiated by the PFs by setting the *GLOBR* bit in the *GLGEN_RTRIG* register. The EMP initiates this reset by setting a bit in an internal register. Global reset is also initiated following a Force TCO command (if it is not disabled by the *Force TCO Reset Disable* bit in the NVM).

EMP Reset (EMPR) – EMPR initializes the resources and data path connected to the EMP including its firmware reload. EMPR is triggered internally by the EMP watchdog timer expiration, by the EMP setting an internal flag due to Force TCO command or due to uncorrectable ECC error in one of the EMP memory shells.

Table 4-1. Device Logic Affected by the Reset Sources

Reset Activation	POR	STRST / PCIR / PERST	EMPR	GLOBR	CORER	PFR/ FLR	VMR	VFR/ VFRLR
Load the NVM to the shadow RAM in the hardware and clear the alternate structure.	+							
Load the EMP firmware from the NVM.	+		+					
Load device settings from NVM shadow and alternate structure (see details listed in Table 6-2 and the reset flow sections that follow).	Shad	Both	Both	Both	Both			
PF MAC addresses (switch and WoL). ¹	+	+ ¹	+	+	+	+		
MAC and PHY interface.	+	¹	+	+				
EMP subsystem including firmware reload.	+		+					
Sticky PCIe context.	+	²						
PCIe HWInit parameters.	+	PE_RST_N only						
PCIe RO registers.	+	+						
PCIe RW/RW1C registers.	+	+				+ ³		VF by VFRLR
Bus master disable. ⁴	+	+	+	+	+	PF	VM	VF
All CSRs - refer to the Programming Interface section for the reset source of each register.								
Most of the PF and VF registers (PF registers are named - PFxxx and VF registers named - VFxxx and VPxxx). Refer to the Programming Interface section for the reset source of each register.	+	+	+	+	+	PF and its VFs		VF only
Cache contexts (filters, queue context) and FPMs settings (sector descriptors and cache entries).	+	+	+	+	+	PF and its VFs	VM	VF
Invalidate VF queue mapping tables (VPLAN_QTABLE).	+	+	+	+	+	VFs of the PF		VF
Invalidate VSI context (including the switch, VSILAN_QTABLE and all other VSI registers).	+	+	+	+	+	PF and its VFs	by the PF SW ⁵	by the PF SW ⁵



Table 4-1. Device Logic Affected by the Reset Sources

Reset Activation	POR	STRST / PCIR / PERST	EMPR	GLOBR	CORER	PFR/ FLR	VMR	VFR/ VFLR
Load the PFs MAC address to the switch and the WOL filters from the NVM (not all PFs must have a WOL filter).	+	+ ⁶	+	+	+	PF		
Tx and Rx data path + Tx scheduler.	+	+	+	+	+	PF and its VFs	VM	VF
Tx and Rx packet buffers.	+	+	+	+	+			
Tx and Rx queue disable.	+	+	+	+	+	PF and its VFs	by the PF SW	VFs
Admin queue disable (POR and EMP also clear the queue context memories).	+	+	+	+	+	PF and its VFs		VF
Disable interrupts.	+	+	+	+	+	PF and its VFs		VF
Interrupt cause control registers.	+	+	+	+	+	By the PF SW	By the PF SW	By the PF SW
RSS key and table.	+	+	+	+	+	PF		VF
Invalidate FD filters.	+	+	+	+	+	PF		

1. PF MAC addresses (switch and WoL) are cleared by hardware at POR and loaded from NVM by the firmware at any of the highlighted reset causes. Following PCIR, the switch MAC addresses are loaded following the assertion of the PCIe reset and the WoL MAC addresses are loaded only following the de-assertion of the PCIe reset. The MAC and PHY are cleared only if the MNG_VETO flags in all four PRTPM_GC registers are cleared. If the WUC filter is inactive and no port is needed by FW/MNG (PRTPM_GC. EMP_LINK_ON is clear), GLOBR is maintained low by hardware during PCIR de-assertion for power savings.
2. Sticky PCIe context is cleared on PERST if no AUX power as documented in [Section 11.2.1](#).
3. For exception list of registers that are not cleared on PFR see [Section 11.2.1](#).
4. The X710-TM4 has several flags that control the Bus Master Enable (BME). BME flags on the PCI configuration space (for each PF and VF) and the VMRD flag in the VSIGEN_RSTAT registers for each VM that is not a VF.
The BME flags of all PFs are cleared by all reset causes indicated by + symbol and a specific PF by FLR.
The BME flags of all VFs are cleared by all reset causes indicated by + symbol and a specific VF by VFLR.
The VMRD flags of all VSIs are set by all reset causes indicated by + symbol and a specific VSI by VMR.
Note that as opposed to BMEs of the function that are cleared by the previous resets (disabling master accesses), the VMRD flags are set (enabling bus master accesses when the BME of their parent PF is set as well).
5. VSIs and its related switch context are cleared by admin command(s) initiated by the software.
6. Note that as opposed to any logic in the X710-TM4 that is initialized at the leading edge of PE_RST_N, the WOL filters are initialized at the de-assertion of PE_RST_N (entering the D0u state).

Table 4-2. NVM Section Loaded by Reset Source Covered by GLNVM_SRLD Register

NVM Section	POR	PERST	PCIR	EMPR	GLOBR	CORER
POR Registers Auto Load	+					
PCIe Analog Configuration	+					
PCIR Registers Auto Load	+	+	+			
PCIe Transaction Layer (TL) Shared	+	+	+			
RO PCIe LCB	+	+				
CORER Registers Auto Load	+	+	+	+	+	+
GLOBR Registers Auto Load	+	1	1	+	+	
PHY Analog Configuration	+					
EMPR Registers Auto Load	+			+		

1. GLOBR registers only auto load if the MNG_VETO flags in all four PRTPM_GC registers are cleared.



4.1.2 Reset Flows

This section describes the reset flows in the X710-TM4 and software interaction.

4.1.2.1 POR Flow

For POR flow see [Section 4.2](#).

4.1.2.2 PCI Reset and Inband PCI Reset Flow

The internal reset flow is shown in [Figure 4-3](#) and described by the text that follows.

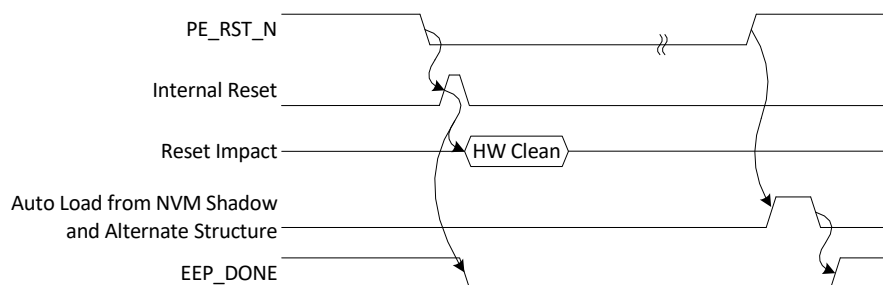


Figure 4-2. PCI Reset Flow

- Avoid any further master accesses on the PCIe bus and discard any completions for the PF.
- Initialize PCIe registers and core registers as listed in [Table 4-1](#).
- Invalidate the LAN queues mapping tables of the PF and its VFs: VFQTABLE's and VSIQTABLE's.
- Invalidate the FPM tables of the PF and its VFs (function private memory).
- Disable all transmit, receive and admin queues. All packets in the X710-TM4 are lost. EMP packets might be lost as well for a very short period (in the range of 1 μ s).
- Invalidate all filters in the FD table and internal caches.
- Invalidate all internal caches: transmit and receive queue contexts.
- Invalidate all associated VSI contexts.
- Clear interrupt settings of all functions.
- The following flags are cleared by the X710-TM4 at the beginning of the reset flow hardware:
 - HW_*_DONE bits in GLNVM_SRLD register (those ones that are listed in [Table 4-2](#) in the respective PCIR or PERST columns).
 - The EMP_*_DONE bits in GLNVM_EMPLD register are cleared (those ones that are listed in [Table 4-2](#) in the respective PCIR or PERST columns) if the respective GLNVM_EMPRQ.EMP_*_REQD bit is set.
 - The CONF_*_DONE bits in GLNVM_ULD register are cleared if the respective HW_*_DONE bit or EMP_*_DONE bit are 0b.



- The X710-TM4 repeats the following flow for all previous *_*_DONE flags:
 - If the NVM is not valid:
 - Set the HW_*_DONE flag in GLNVM_SRLD register.
 - If the NVM is valid:
 - Load the matched block from the shadow memory and set the HW_*_DONE flag
 - If EMP_*_REQD is set:
 - Load any parameters from the alternate structure into the matched block(s)
 - Perform any configuration of the matched block(s)
 - Set the EMP_*_DONE flag
 - For CORER and GLOBR modules, once HW_*_DONE and EMP_*_DONE flags are set, the matched CONF_*_DONE flag is set as well. For other modules, once HW_*_DONE flag is set, the matched CONF_*_DONE flag is set as well.
 - At this point, the EMP might start responding the Get Version Admin command, which is blocked until this stage.
 - When checking for hardware configuration to complete, software should either wait for a response to a Get Version Admin command or poll on the CONF_*_DONE bits.

Following these steps, hardware is ready to accept operating system configuration cycles.

4.1.2.3 PFR Flow

PFR resets a specific PF. For SR-IOV, it also resets the VFs of this PF. The reset flow is shown in [Figure 4-4](#). It is initiated by the PF driver (setting the *PFSWR* bit in the *PFGEN_CTRL* register) or operating system (setting the *FLR* flag in the Device Control register) or D0 to D3hot transition.

Before initiating the PFR, software is expected to disable all transmit and receive queues of the PF as described in the following pseudo code:

```
disable_pf_queues() // disable all Tx and Rx queues of the PF
```

```
{
```

1. Disconnect all Tx and Rx queues from the interrupt link lists as follows:
 - Set the *FIRSTQ_INDX* field to 0x7FF in all *PFINT_LNKLSTx* register of the PF.
 - Set the *FIRSTQ_INDX* field to 0x7FF in all *VPINT_LNKLSTx* register of the VFs of the PF.
2. `last_q = PFLAN_QALLOC.LASTQ - PFLAN_QALLOC.FIRSTQ` // `last_q` is a local variable used in the steps that follow. It is the index of the last queue of the PF and its VFs (in the PF space)
3. Set the *SET_QDIS* flag in the *GLLAN_TXPRE_QDIS* registers for all transmit queues of the PF and its VFs. The queue indexes are global ones starting by *PFLAN_QALLOC.FIRSTQ* and up to *PFLAN_QALLOC.LASTQ*.
4. Clear *QRX_ENA*[register index=0,... last_q] // Disable all Rx queues of the PF.
5. Clear *QTX_ENA*[register index=0,... last_q] // Disable all Tx queues of the PF. Software must guarantee a gap of at least 400 μs from step 3 to this step.
6. Wait for the last Rx queue to be disabled and wait for all Tx queues to be disabled or time expires (expiration time is defined by the *ENDLESS_XOFF_THRESH* parameter). Note that if software is not required to take any special actions if time has expired and can continue to the next step initiating the PFR.

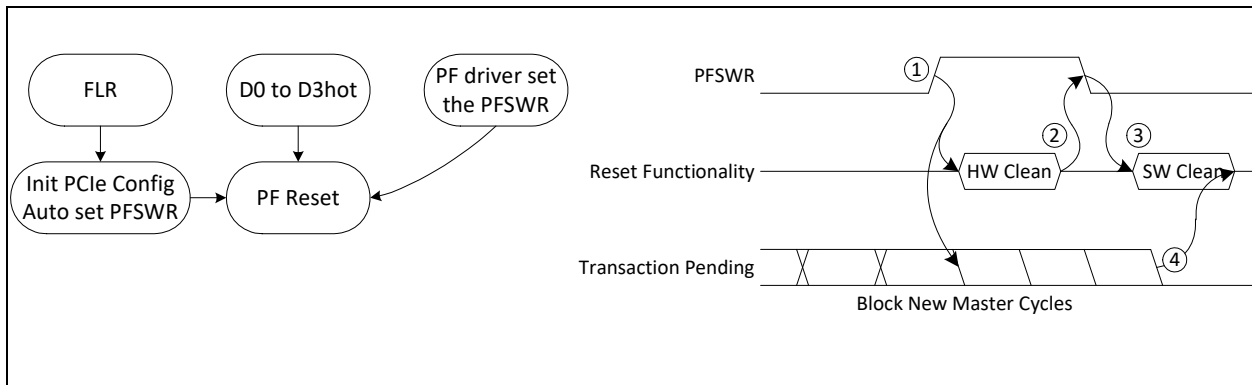


Figure 4-4. PFR Flow

The PF hardware response to PFR is listed in [Table 4-1](#). Note to the following specific events:

- Emulate internal VFR to all its VFs (the hardware response to VFR is described in [Section 4.1.2.5.3](#) with the exception that the VF’s CSRs are not gated on read).
- Avoid any further master accesses on the PCIe bus and discard any completions for the PF and its VFs.
 - Once all pending requests of the PF complete, the *Transaction Pending* bit in the Device Status register in the PCIe configuration space is cleared.
 - Once all pending requests of each VF of the PF completes, the *Transaction Pending* bit in the Device Status register the VF PCIe configuration space is cleared (per each VF).
- Disable all transmit receive and admin queues of the PF and its VFs (note that some transmit packets that are already fetched completely might be transmitted).
 - Disabling the transmit queues might take some time until the X710-TM4 processes all transmit descriptors that are already fetched. Because many queues are active, this process might take longer. The transmit completion might be further delayed when the TC is paused by flow control. It can happen if other functions have active transmit queues on the same TC as the function under reset. Hardware includes a timeout mechanism that prevents indefinite latency as described in [Section 7.7.1.2.8](#).
 - Disabling the receive queues might take some time as well until the packets of the function’s queues are flushed from the shared receive data path. The completion might be further delayed when there are packets of other functions in the pipe that belongs to no-drop TC with no valid receive descriptors. Hardware includes a timeout mechanism that prevents indefinite latency as described in [Section 7.7.1.2.8](#).
- Clear interrupt settings of the PF and its VFs (excluding the CEQ and LAN transmit and receive cause control registers and any interrupt context in the PCIe configuration space). Write also 1b to clear the *VFLRE* flags in the *GLGEN_VFLRSTAT* registers of all the VFs owned by the PF.

After all the previous steps complete, hardware does the following:

- Clears the *PFSWR* bit in the *PFGEN_CTRL* register.
- Releases bus master cycles for the PF and its VFs.

PF software polls the *PFSWR* bit until cleared (indicating that hardware completed its reset flow). On top of it, software should also poll the *Transactions Pending* flag in the PCI configuration space of the PF (indicating that all outstanding requests of the PF completed).

- Once the *Transactions Pending* flag clears, PF software can release the pinned memory structures.



- Once the *PFSWR* bit clears as well, PF software can proceed to the following steps.

Software Note: While polling for the *PFSWR* bit to be cleared, software should also poll the *DEVSTATE* field in the *GLGEN_RSTAT* register for at least 200 ms. If this field is non-zero, software should abandon the PFR flow and prepare for the upcoming global reset.

As part of the initialization flow, the PF driver checks the pending transactions of its VFs (by setting the VF index and the offset of the VF Device Status register in the *PF_PCI_CIAA* register and then polling the pending flag in the *PF_PCI_CIAD* register). Once the transactions pending is found cleared, PF software does the following:

- Clears the *VFSWR* flag in the *VPGEN_VFRTRIG* registers of its VFs.
- Sets the *VFR_STATE* in the *VFGEN_RSTAT* registers of its VFs to VFR completed. At this point, the memory structures of the VF can be released and the VF software can start its initialization flow.

As part of the software/hardware initialization flow, the PF software should check for potential race condition with another PF or EMP initiating a global reset (*CORER*, *GLOBR* or *EMPR*):

- Query the reset counters (*CORERCNT*, *GLOBRCNT* and *EMPRCNT*) in the *GLGEN_RSTAT* register.
- Interrupt enable flow.
- Read the *GLGEN_RSTAT* register for the device state (*DEVSTATE*) and the reset counters (*CORERCNT*, *GLOBRCNT* and *EMPRCNT*).
 - If *DEVSTATE* = Reset requested or Reset in progress then go to the global reset flow.
 - If Device state = Device active.
 - If the updated values of the global reset counters equal the value sampled at the beginning of this procedure then all good. Software can continue with the rest of the software/hardware initialization flow.
 - Else, then go to the global reset flow.

4.1.2.4 FLR Flow

FLR resets a specific PF. In the case of SR-IOV, it also resets the VFs of this PF.

- The following steps are optional:
 - The operating system is expected to clear the *BME* bit in the Command register in the PCI configuration register of the PF.
 - For SR-IOV, the operating system is expected to also clear the *BME* bit in the VF Command register in the PCI configuration register of all VFs of this PF.
 - As a response, hardware avoids any new master cycles of the PF and its VFs (including MSI-X initiation).
 - The operating system should poll the *Transaction Pending* bit in the Device Status register of the PF until cleared.
 - For SR-IOV, the operating system should poll also the *Transaction Pending* bit in the VF Device Status register of all VFs of the PF until cleared.
 - Note that all the previous steps are expected and recommended but not enforced by the PCI specification.
- The operating system sets the *FLR* bit in the Device Control register of the PF. The operating system is required by PCIe specification to wait 100 ms before it can assume that the FLR sequence completes by hardware.
- The PF hardware response as follows:
 - Initialize the PCIe configuration space of the PF including clearing the *BME* bit of the PF and the *VFE* bit.



- By clearing *BME*, hardware avoids any further master accesses on the PCIe bus and trash any completions for the PF.
 - By clearing the *VFE* bit, all VFs of the PF avoid any further master accesses on the PCIe bus and discards any completions targeted for these VFs and become hidden on the PCIe bus.
 - As part of the PCIe configuration initialization, the completion timeout is set to its hardware default.
 - Once all pending requests of the PF and its VFs complete or completion timeout expires (whichever comes first), the *Transaction Pending* bits in the PCIe configuration space are cleared.
- Auto set the *PFSWR* bit in the PFGEN_CTRL register (triggering a PFR described in the section that follows).

Once the *Transaction Pending* bit in the PCIe configuration space of each VF clears, the operating system can release all VF memory structures and unload the VF driver (if required). Once the *Transaction Pending* bit in the PCIe configuration space of the PF clears, the operating system can release all PF memory structures and unload the PF driver (if required).

4.1.2.5 VFR/VFLR Flows

The VF reset flow is shown in Figure 4-5 and detailed in the sections that follow.

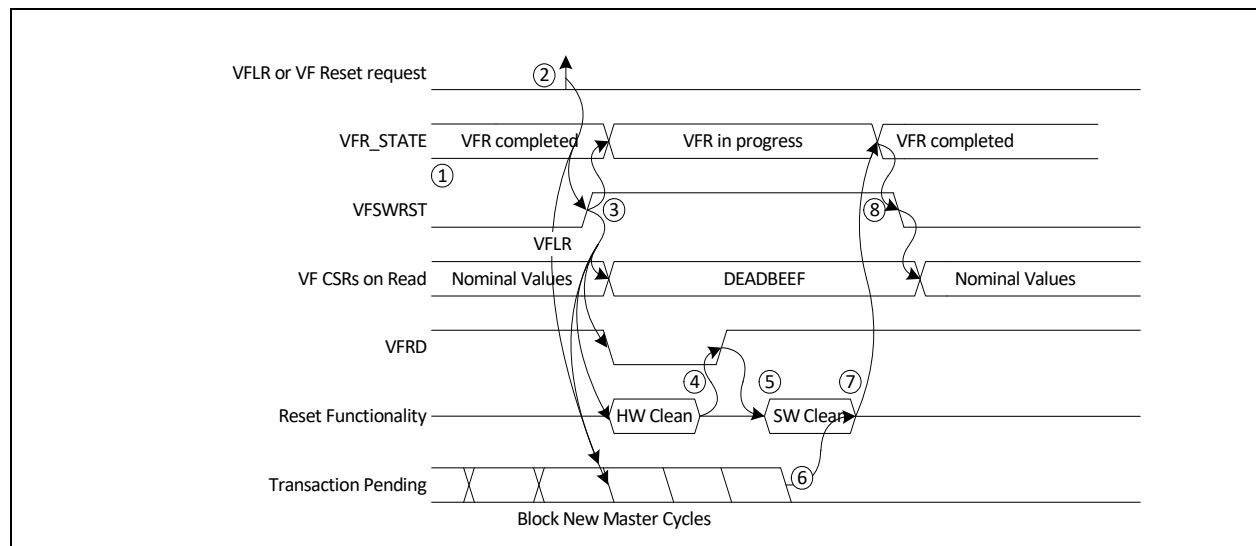


Figure 4-5. VF Reset (VFR) Flow

4.1.2.5.1 VF Reset Request by the VF Driver

The VF driver requests the VF reset from its parent PF as follows and is shown in Figure 4-5. See also Section 4.1.2.5.3 that describes the PF response to the VF reset request.

- The *VFR_STATE* in the *VFGEN_RSTAT* register in this step is expected to be *VFR completed*. Note that the VF software gets a control over its VF only after any prior VF reset flow completes (step 1 in Figure 4-5).

- The VF driver initiates a request to its parent PF to initiate a VF reset. Note that the mechanism for sending this request is outside the scope of this document. It could be done using a VF-to-PF mailbox (admin command) or any other software-based sideband channel (step 2 in [Figure 4-5](#)).
- After that, the VF polls the VFR_STATE in the VFGEN_RSTAT register until set by the PF-to-VFR completed (step 8 in [Figure 4-5](#) and explained in [Section 4.1.2.5.3](#)).
- The VF software proceeds activating the function.

4.1.2.5.2 VF Reset Request by the Operating System (VFLR)

The VF reset initiation by the operating system is described as follows and shown in [Figure 4-5](#).

- The following steps are optional:
 - Clear the *BME* bit in the VF Command register.
 - As a response, hardware avoids any new master cycles of the VF (including MSI-X initiation). Old completions are trashed by hardware.
 - The operating system polls the *Transaction Pending* bit in the VF Device Status register until cleared.
- The operating system sets the *FLR* bit in the VF Device Control register (step 2 in [Figure 4-5](#)).
- The operating system is required by PCIe specification to wait 100 ms before it can assume that the VFLR sequence is completed by hardware.
- If required, the operating system brings up a new VF (or the same VF). The VF software driver should poll the VFR_STATE in the VFGEN_RSTAT register until it equals to VFR completed (set by the PF software).
- The VF software proceeds activating the function.

Hardware responses to VFLR are as follows:

- Initialize the PCIe configuration space of the VF including the *Bus Master Enable* flag. The *Transaction Pending* bit is cleared when there are no more pending completions.
 - Once the *Transaction Pending* bit in the VF Device Status register is cleared, the operating system can release all VF memory structures and unload the VF driver (if required).
- Set internally the *VFSWR* bit in the VPGEN_VFRTRIG register of the VF, which initiates a VFR (described in [Section 4.1.2.5.3](#)).
- Set the matched *VFLRE* flag in the GLGEN_VFLRSTAT registers and initiate an interrupt to the parent PF. The PF response to the VFLR interrupt is described in [Section 4.1.2.5.3](#).

4.1.2.5.3 VF Reset Flow by the PF Software Driver

PF software is called to initiate the VFR by the VF software driver or as a result of a VFLR interrupt. See [Figure 4-5](#).

- Specific step for a VFLR interrupt:
 - The PF software queries the *VFLRE* flags in the GLGEN_VFLRSTAT registers (of the VFs that it owns).
- Specific step for a VF reset request by the VF driver:
 - PF software sets the *VFSWR* bit in the VPGEN_VFRTRIG register of the VF (step 3 in [Figure 4-5](#)).

Hardware response to setting the *VFSWR* bit is listed in [Table 4-1](#). Note that as part of the VF registers, the VFGEN_RSTAT register is cleared as well reflecting VFR in progress state. On top of it, hardware also gates read accesses to the CSRs of the VF returning DEADBEEF or DEADBEAF values and also gates any master accesses.



- When the reset flow completes, hardware sets the *VFRD* flag in the *VPGEN_VFRSTAT* register and sets the matched *VFLRE* flag in the *GLGEN_VFLRSTAT* registers.

Software Note: If a timeout occurs while polling for the *VPGEN_VFRSTAT.VFRD* bit, retry the reset:

- Clear *GLGEN_VFRTRIG.VFSWR*.
- Set *GLGEN_VFRTRIG.VFSWR*.
- Restart the polling for *VPGEN_VFRSTAT.VFRD*.

If several retries do not prevent the timeout, treat it as an error. Escalating to a PFR might be a good idea in some cases.

Once the *VFRD* flag in the *VPGEN_VFRSTAT* register is found active, the PF software proceeds with the VF reset flow (step 5 in [Figure 4-5](#)).

- Writes 1b to clear, to the matched bit in the *GLGEN_VFLRSTAT* registers for the VF under reset. Note that the *GLGEN_VFLRSTAT* registers are composed of 128 bits for the 128 VFs. All PFs have access to the bits of all VFs. However, it is expected that the PFs writes 1b to clear to those bits that match its VFs and only to those bits.
- Disable the receive queues of the VF following the fast queue disable flow per each queue as described in [Section 8.3.3.1.3](#). Note that hardware auto disables the transmit queues of the VF.
- Clear the interrupt settings of the VF.
- Clear the queue mapping tables of the VF's VSIs (*VSIQTABLES*).
- Clear the filters in the FD table that were assigned by the PF for the VF.
- Remove all the MAC/VLAN filters from the VSIs of the VF and then remove these VSIs.
- The PF driver checks the pending transactions of its VF by setting the VF index and the offset of the VF Device Status register in the *PF_PCI_CIAA* register and then polling the pending flag in the *PF_PCI_CIAD* register (step 6 in [Figure 4-5](#)).
- The following steps in this bullet item are part of re-enabling the VF. It can be executed at this phase before notifying the VF that the reset flow completed or at a later phase (depending on software implementation):
 - Add the VSIs for the VF (including its transmit and receive queues and its scheduler).
 - Add the MAC/VLAN filters for the VSI.
 - Enable the *VFLAN_QTABLE* by setting the *VPLAN_MAPENA* and then program the *VFLAN_QTABLE* to the queues of the VF.
- PF software completes the flow by notifying the VF that the reset flow completed. It sets the *VFR_STATE* in the *VFGEN_RSTAT* register to *VFR completed* and clears the *VFSWR* bit in the *VPGEN_VFRTRIG* register (step 7 in [Figure 4-5](#)).

Hardware response to a cleared *VFSWR* flag:

- VF CSRs are unlocked. As a result, the VF software can see the updated *VFR_STATE* in the *VFGEN_RSTAT* register that equals to *VFR completed* (step 9 in [Figure 4-5](#)).
- Master cycles are not blocked anymore by the reset logic.
- Hardware is ready to be used by the VF.

4.1.2.6 VMR Flow

Before initiating a VM reset, the PF software should disable the interrupts associated with this VM. The VM interrupt causes might share the same interrupt with the PF. In this case the PF software need only to remove the VM interrupts causes from those interrupt linked list as described in [Section 7.5.3.1.3](#). Then the PF software can initiate the VM reset and re-enable the interrupts of the PF.



The PF sets the *VMSWR* bit in the *VSIGEN_RTRIG* register:

- Hardware response to VMR is the same as its response to VFR (other than invalidating the *VFQTABLE*'s and *VFxxx* registers, which the VM does not own).

Software Note: If a timeout occurs while polling for the *VSIGEN_RSTAT.VMRD* bit, retry the reset:

- Clear *VSIGEN_RTRIG.VMSWR*.
- Set *VSIGEN_RTRIG.VMSWR*.
- Restart the polling for *VSIGEN_RSTAT.VMRD*.

If several retries do not prevent the timeout, treat it as an error. Escalating to a PFR might be a good idea in some cases.

The PF driver polls the VMR done indication in the *VSIGEN_RSTAT* register and then executes the following complementary steps (as the VFR flow):

- Disable the receive queues using the fast queue disable flow. Note that hardware auto disables the transmit queues of the VM.
- Optionally clear the queue mapping tables, FD filters and VSI context.
- The PF polls the *Transactions Pending* flag of the VM verifying that there are no transaction pending of the VM as follows: Set the VSI index in the *PFPCI_VMINDEX* and then poll the *PFPCI_VMPEND* register.
- Mimic the response for VF reset for the VM resources other than the following changes:
 - No clearing the interrupts settings that might be shared with the PF (or other VMs).
 - The PF driver does not check a pending transactions of the VM which does not exist.
- The PF completes the flow by clearing the *VMSWR* bit in the *VSIGEN_RTRIG* register.

4.1.2.7 Core Global and EMP Reset Flows

The global resets can be initiated by the PF software or the EMP firmware. It is expected to be used as a mechanism to resolve potential hardware locks or synchronization lose, bringing the X710-TM4 to a known functional state. These global resets impact all PFs (and their VFs) as well as the EMP subsystem (EMP reset only). Therefore, a graceful flow is enabled by hardware as shown in [Figure 4-6](#) and described in the following sections. Software initiates the global resets using the *GLGEN_RTRIG* register. The EMP can access the same register or use an internal register.

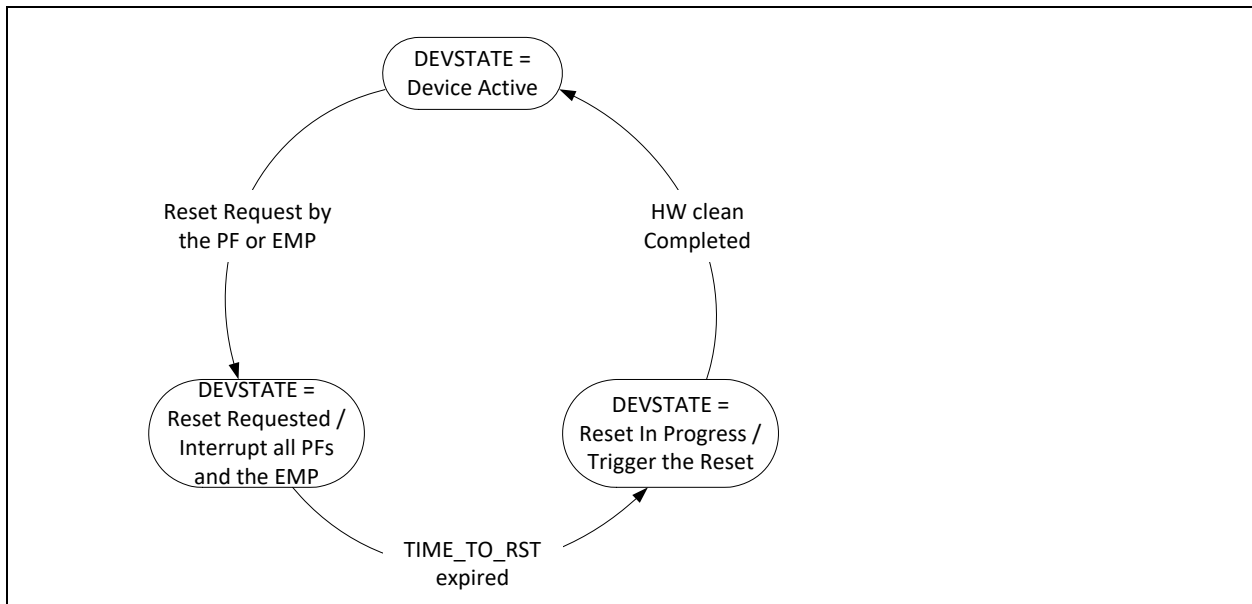


Figure 4-6. Global Resets Flow

4.1.2.7.1 Software and Firmware Interface

GLOBR / CORER — Setting the *GLOBR* flag or *CORER* flag in the *GLGEN_RTRIG* register triggers a graceful global/core reset, respectively.

EMPFWR — EMP firmware watchdog expiration or EMP setting internal EMP reset flag triggers a graceful EMP reset (EMPR).

DEVSTATE — Global device state exposed to all PFs in the *GLGEN_RSTAT* registers. *DEVSTATE* can be at one of the following states: device active, reset requested or reset in progress.

RESET_TYPE — *RESET_TYPE* reflects the last reset cause initiated to the X710-TM4. It changes its state once any of the following reset sources are triggered. *RESET_TYPE* can be at one of the following states: POR, CORER, GLOBR and EMPR.

RST_CNT — The *GLGEN_RSTAT* reflects the following counters: *CORERCNT*, *GLOBRCNT* and *EMPRCNT*. These fields are 2-bit counters each. They count the matched reset completion events since POR.

GRSTDEL — *GRSTDEL* in the *GLGEN_RSTCTL* register is the delay from reset request to its initiation by hardware. *GRSTDEL* is loaded from the NVM and defined in 100 ms units up to ~6.5 seconds.

TIME_TO_RST — *TIME_TO_RST* is a down counter, loaded by the *GLGEN_RSTCTL.GRSTDEL* register following a reset request. When *TIME_TO_RST* reaches a zero value, hardware initiates the requested reset.

4.1.2.7.2 CORER Flow

PF software or EMP firmware initiates a graceful core reset by setting the *CORER* flag in the *GLGEN_RTRIG* register and polling the X710-TM4’s state in the *GLGEN_RSTAT* register.

Hardware response:



- Changes the GLGEN_RSTAT register are as follows:
 - Set DEVSTATE to reset requested.
 - Set RESET_TYPE to CORER indicating the requested reset.
 - Set TIME_TO_RST by the GLGEN_RSTCTL.GRSTDEL value and start counting down.
- Further write accesses to the GLGEN_RTRIG register do not re-trigger the TIME_TO_RST. Still, if a stronger reset is set, it overrides this reset and is reflected in the RESET_TYPE field.
- Initiates a GRST interrupt to all PFs and EMP that the reset is about to be fired by hardware.
- Once the GLGEN_RSTAT.TIME_TO_RST gets to zero, hardware triggers the internal core reset and changes the GLGEN_RSTAT.DEVSTATE to reset in progress.
- The following flags are cleared by the X710-TM4 at the beginning of the reset flow.
 - HW_*_DONE bits in GLNVM_SRLD register (those ones that are listed in Table 4-2 in the respective PCIR or PERST columns) are cleared.
 - The EMP_*_DONE bits in GLNVM_EMPLD register are cleared (those ones that are listed in Table 4-2 in the respective PCIR or PERST columns) if GLNVM_EMPRQ.EMP_*_REQD is set.
 - The CONF_*_DONE bits in GLNVM_ULD are cleared if the respective HW_*_DONE bit or EMP_*_DONE bit are set to 0b.
- Clear the entire transmit and receive data path, avoid any further master accesses on the PCIe bus, discard any completions for the entire device and abort any transmission in progress (including packets sent by the EMP).
- Reset internal device logic excluding EMP cluster, PCI interface and MAC/PHY cluster as listed in Table 4-1.
- Once the reset flow completes, update the GLGEN_RSTAT register as follows:
 - Set DEVSTATE to device active.
 - Increment the CORERCNT by one.
- The X710-TM4 repeats the following flow for all previous *_*_DONE flags:
 - If the NVM is not valid:
 - Set the HW_*_DONE flag in GLNVM_SRLD register.
 - If the NVM is valid:
 - Load the matched block from the shadow memory and set the HW_*_DONE flag
 - If EMP_*_REQD is set:
 - Load any parameters from the alternate structure into the matched block(s)
 - Perform any configuration of the matched block(s)
 - Set the EMP_*_DONE flag
 - Load the matched block from the shadow memory and set the HW_*_DONE flag
 - For CORER and GLOBR modules, once HW_*_DONE and EMP_*_DONE flags are set, the matched CONF_*_DONE flag is set as well. For other modules, once HW_*_DONE flag is set, the matched CONF_*_DONE flag is set as well.
 - At this point, the EMP might start responding the Get Version Admin command, which is blocked until this stage.

Following the GRST interrupt, all PFs and the EMP poll the X710-TM4's state in the GLGEN_RSTAT register.

- Keep polling the register as long as DEVSTATE is not equal to device active. Note that as long as DEVSTATE equals to reset requested, the TIME_TO_RST indicates the remaining delay to the internal reset triggering by hardware. Note the PF drivers should avoid any CSR slave accesses other than the one required to query the X710-TM4 state.



- The GLGEN_RSTAT also indicates the number of the initiated global resets via the GLGEN_RTRIG register (CORER, GLOBR, EMPR). These counters might be needed in case a function initiated consecutive global reset before all other functions had the chance to realized that the previous reset completed. It would be good practice to avoid too frequent global resets to avoid such cases.
- Check the RESET_TYPE that indicates the reset that was initiated and follow the required initialization flow.

Note: The function that initiated the reset should check the RESET_TYPE since it might reflect a stronger reset initiated by another function.

- Check that hardware completed auto loading its settings from the NVM shadow and the alternate structure by polling the CONF_*_DONE flags in GLNVM_ULD register.
 - When checking for the hardware configuration to complete, software should either wait for a response to a Get Version Admin command or poll on the CONF_*_DONE bits.
- The PFs proceeds with their software initialization flow.

4.1.2.7.3 GLOBR Flow

Global reset is initiated by the PFs or the EMP by setting the *GLOBR* flag in the GLGEN_RTRIG. The GLOBR flow is identical to the CORER flow with the following changes:

- GLOBR also initializes the MAC/PHY units.
- GLGEN_RSTAT.RESET_TYPE is set by hardware to GLOBR (rather than CORER).
- Increment the GLOBRCNT by one (rather than CORERCNT).
- Loading the X710-TM4's settings from the NVM is listed by the GLOBR column in [Table 4-2](#).

4.1.2.7.4 EMPR Flow

EMP reset is expected to be used by the EMP as a mechanism to resolve potential hardware locks or potential lost of synchronization between firmware and hardware that are not expected to be resolved by CORER nor by GLOBR. EMPR impacts also all PFs and their VFs, therefore the following graceful flow is recommended.

EMPR flow is identical to the CORER flow with the following changes:

- During normal operation, the EMPR can be initiated only by the EMP by setting an internal EMP reset flag.
- EMPR also initializes the MAC/PHY units as well as the EMP cluster.
- GLGEN_RSTAT.RESET_TYPE is set by hardware to EMPR (rather than CORER).
- Increment the EMPRCNT by one (rather than CORERCNT).
- Loading the X710-TM4's settings from the NVM is listed by the EMPR column in [Table 4-2](#).



4.2 Cold Reset Initialization

This section describes the flow of the X710-TM4 power up following a cold reset (application of power to the X710-TM4). The initialization sequence for the X710-TM4 is broken down into phases: power on, BIOS initialization, and software device driver load.

- Power on ([Section 4.2.1](#)) is the first phase that includes all steps required to support pre-boot power management and manageability, satisfy the PCIe requirements for exiting cold reset, and prepare for the BIOS initialization phase. This stage also covers firmware initialization.
- BIOS initialization ([Section 4.3](#)) begins when option ROM code is loaded by BIOS in order to provide boot services for PXE or iSCSI. Other option ROM capabilities include configuration images for certain OEM environments such as SMASH/CLP.
- The final phase of the X710-TM4 initialization is software device driver load, where the operating system loads the various device drivers and the X710-TM4 has been initialized for its post-boot operation. See [Section 4.4](#).

4.2.1 Power On

[Figure 4-7](#) and [Figure 4-8](#) describe the stages that make up the power up sequence. Note that two procedures take place in parallel following initialization of internal clocks and loading hardware-managed units from the NVM:

- The on-die processors are initialized, starting with the EMP
- Once PE_RST_N is de-asserted, the PCIe link goes through its initialization flow

[Figure 4-7](#) shows Case I, where APM or pass-through manageability are enabled at pre-boot. In that case, the Ethernet link is brought up once the EMP has initialized, independent of the de-assertion of PE_RST_N.

[Figure 4-8](#) shows Case II, where APM and pass-through manageability are both disabled pre-boot. In that case, the Ethernet link kept down as long as PE_RST_N is kept asserted. Once PE_RST_N is de-asserted, the Ethernet link is brought up (but not before EMP completed initialization).

Note: In both cases, the Ethernet link is brought up only after the EMP completes its initialization.

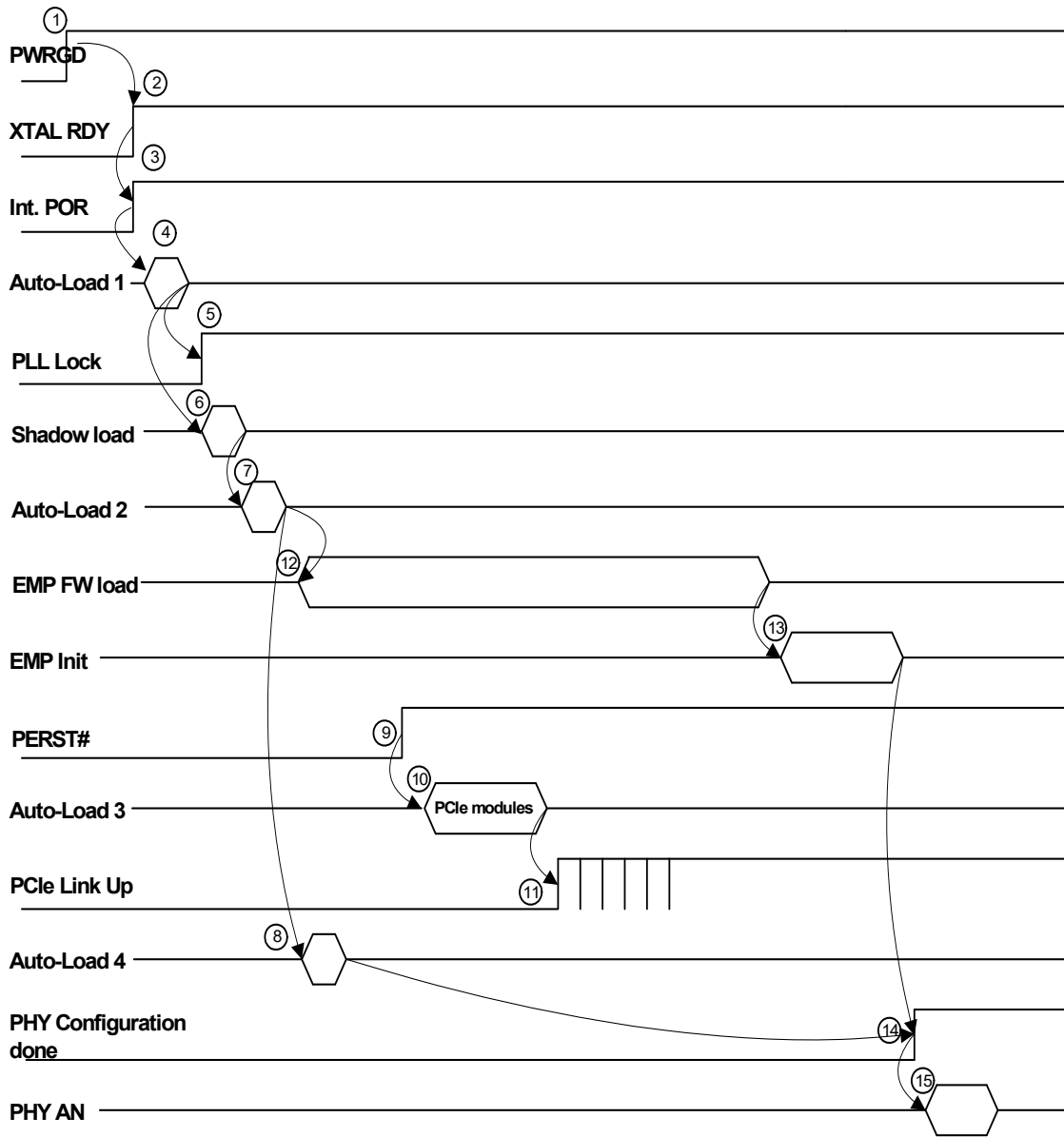


Figure 4-7. Power-on Sequence - Case I

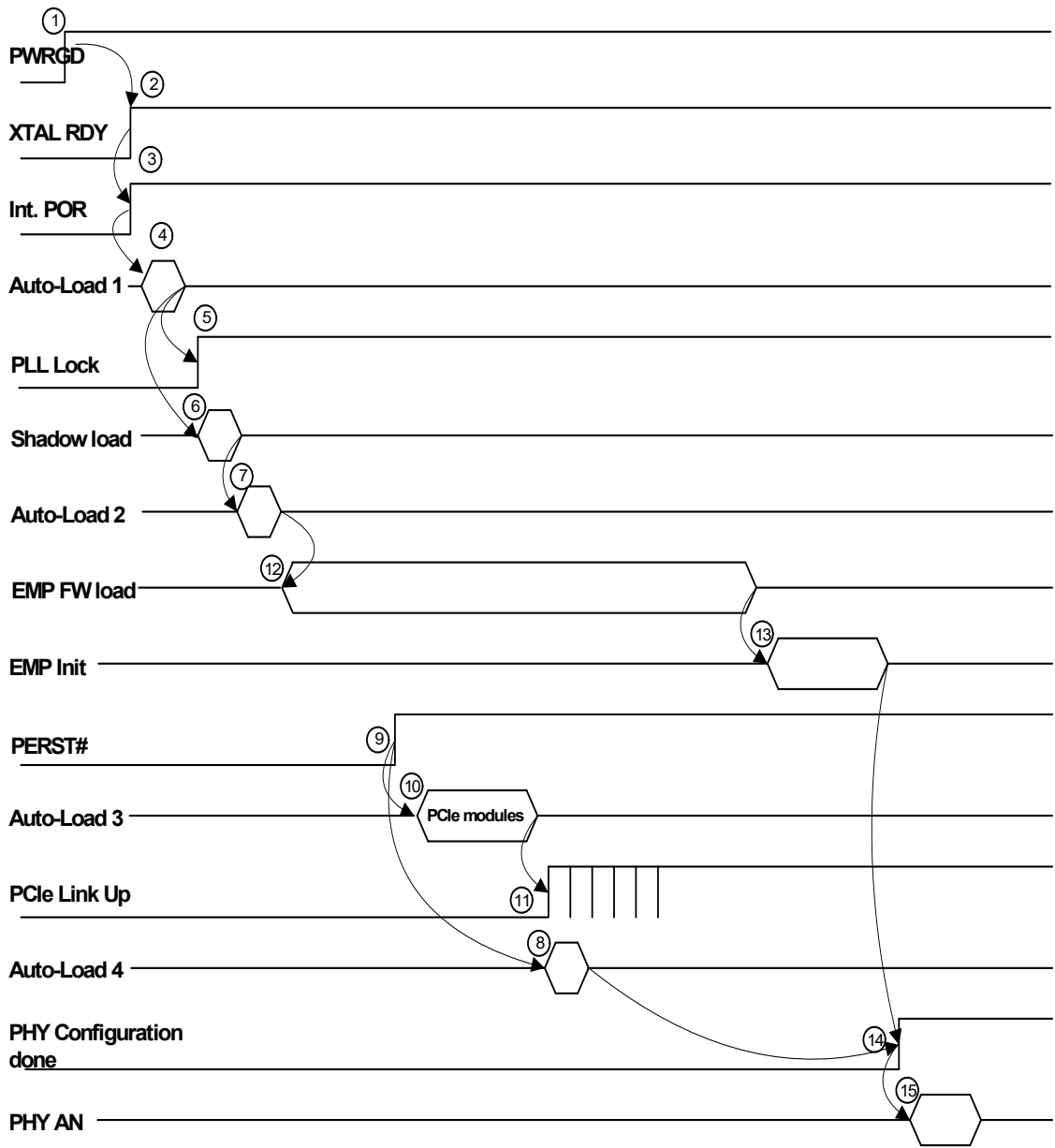


Figure 4-8. Power-on Sequence - Case II



Table 4-3 lists the relationship between steps and the timing of each stage.

Legend:

- Triggering event — This is the event that starts the corresponding step.
- Duration — Time it takes to complete the respective step (usually defines the maximum duration).
- Completion Time from Start — The time the respective step ends, counting from beginning of the power-up sequence. It usually defines the latest time this step might end.

Example: EMP firmware load

- Step — EMP firmware load
- Triggering event — Auto-load 2 complete (an earlier step)
- Duration — it might take up to 380 ms to complete this step
- Completion Time from Start — Auto load 2 ends at time 79 ms. With this step taking up to 380 ms, completion would be maximum of 459 ms from start

Table 4-3 Power-on Sequence Stages

	Step	Triggering Event	Duration (ms)	Completion Time From Start (ms)
1	Power is applied to the X710-TM4.	-	0	0
2	Xtal clock ready.	Power ramp	35	35
3	Internal POR goes active following clock.	Clock stable	0	35
4	Auto load 1 completes - NVM validity is checked and NVM contents are loaded into the analog sections (PCIe and internal PHY).	Internal POR	21	56
5	The internal PLL is up.	Auto load 1 complete	2	58
6	NVM contents are loaded into shadow RAM.	Internal PLL locked	21	79
7	Auto load 2 completes - POR and EMP modules are loaded from shadow RAM.	Shadow RAM ready	~0	79
8	Auto load 4 completes - units on core and MAC/PHY clock are loaded from shadow RAM. Case I - If APME or manageability is enabled, it is done following auto load 2. Case II - If both APME and manageability are disabled, it is done following PE_RST_N de-assertion. Note: The order between auto load 3 and 4 is not deterministic.	Auto load 2 complete/ PCIe PLL is up	~20	99/130
9	PE_RST_N is de-asserted.	Power ramp	0	≥100
10	PCIe PLL is up. The PCIe unit is loaded from NVM shadow RAM (auto load 3) and PCIe enters a detect state.	PE_RST_N de-assert	5	105
11	PCIe specification requirement - PCIe link must enter a detect state.	PE_RST_N de-assert	20	120
	PCIe specification requirement - PCIe link is ready to process configuration cycles.	PE_RST_N de-assert	100	200
12	EMP firmware load - both manageability and other EMP code.	Auto load 2 complete	380	~460



Table 4-3 Power-on Sequence Stages

	Step	Triggering Event	Duration (ms)	Completion Time From Start (ms)
13	EMP firmware initialized.	EMP firmware load completed	20	480
14	MAC/PHY configuration by EMP is done (either internal PHY or external PHY) and can bring up link (GLNVM_ULD.CONF_GLOBAL_DONE is set).	EMP firmware initialized	~0	480
15	Ethernet link established	MAC/PHY completes configuration	See Note ¹	

1. Specific to the PHY media used.

4.2.1.1 LAN_PWR_GOOD Support

It is possible to begin device initialization based on the assertion of the LAN_PWR_GOOD input rather than based on main power ramp. When the POR_BYPASS input pin is set to 1b, the X710-TM4 disables the internal POR circuit and uses the LAN_PWR_GOOD pin as a POR indication.

Note: LAN_PWR_GOOD should be asserted during pre-boot time (before the operating system boots).

Section 4.2.1.2 and Figure 4-9 depict the initialization flow. Other events listed in Table 4-4 are shifted relative to internal POR and PE_RST_N.

Table 4-4. Power-on Sequence With a LAN_PWR_GOOD Signal

	Step	Triggering Event	Duration (ms)	Completion Time From Start (ms)
1	Power Ramp - Must meet the sequence defined in Section 13.3.1.1. Time = 0 ms is when all power rails are at 90% of nominal voltage.	Power ramp	0	0
2	LAN_PWR_GOOD - Must not be asserted before specified duration.	Power ramp	40	40
3	Xtal clock ready.	LAN_PWR_GOOD	1	41
4	Internal POR goes active following Xtal ready.	Xtal clock ready	0	41
5	PE_RST_N must not be de-asserted before specified duration. Note: The PCIe specification still holds that PE_RST_N is de-asserted at least 100 ms following power ramp.	LAN_PWR_GOOD	50	91

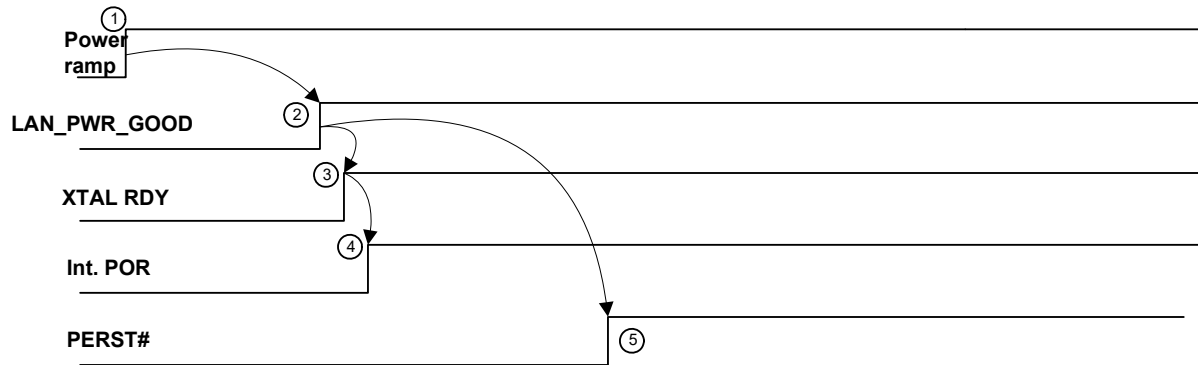
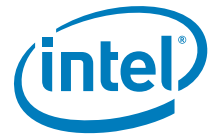


Figure 4-9. Power-on Sequence With a LAN_PWR_GOOD Signal

4.2.1.2 Auto Load Shadow RAM

This phase of auto load determines if a properly-configured Flash device is attached:

- If the attached Flash device is not properly configured, the default hardware settings are kept. The power-on flow continues without loading from the NVM.
 - The NVM is expected to be reprogrammed. See [Section 3.4.4.2](#) for more detail on how this is done.
- If the attached Flash device is properly configured, auto load proceeds with its first stage. NVM modules loaded to the on-die shadow RAM are read from the NVM.

Once the shadow RAM is ready, the GLNVM_GENS.FL_AUTO_RD bit is set. The bit is also set when the NVM is found blank.

4.2.1.3 Auto Load Into Device Units

This stage loads the NVM configuration into the device units, either directly from the NVM or indirectly through shadow RAM.

Loading the following NVM modules is tracked and reported as:

- PCIe analog
- PHY analog
- PCIR registers auto load
- RO PCIe LCB
- PCIe Transaction Layer (TL) shared
- PCIe ALT auto load
- CORER registers auto load
- GLOBR registers auto load



- POR registers auto load
- EMPR registers auto load

The following CSR fields track the progress of loading the NVM modules following power on and the various resets:

- GLNVM_GENS.FL_AUTO_RD, when set, indicates that the shadow RAM was loaded from the NVM and is ready for use.
- GLNVM_SRLD.HW_*_DONE bits (one per relevant module previously identified), when cleared, indicate that the respective module needs to be loaded from shadow RAM into the X710-TM4. When set, it means that the module is not required to load or has already been loaded.
- EMP_*_REQD bits in the GLNVM_EMPRQ register (one bit per relevant module) indicate whether the EMP involvement is required during the initialization of the module.
- EMP_*_DONE bits in the GLNVM_EMPLD register (one bit per relevant NVM module) indicate when the corresponding module completed initialization by the EMP.
- GLNVM_ULD.CONF_*_DONE bits (one per module previously identified), when set, indicate that the respective units are initialized and ready for use.

Default hardware values:

- GLNVM_GENS.FL_AUTO_RD = 0b
- GLNVM_SRLD.HW_*_DONE = 0b
- GLNVM_EMPRQ.EMP_*_REQD = 0b
- GLNVM_EMPLD.EMP_*_DONE = 0b
- GLNVM_ULD.CONF_*_DONE = 0b

The flow during a power-on stage is as follows:

- If the NVM is found blank:
 - All GLNVM_SRLD.HW_*_DONE bits and all GLNVM_ULD.CONF_*_DONE bits are set by the X710-TM4
- If the NVM is found valid:
 - GLNVM_GENS.FL_AUTO_RD is set when the shadow RAM was loaded from the NVM.
 - GLNVM_EMPRQ.EMP_*_REQD is loaded from the NVM.
 - When a module completes loading from the shadow RAM to the X710-TM4, it sets the respective GLNVM_SRLD.HW_*_DONE bit. If the respective GLNVM_EMPRQ.EMP_*_REQD bit is 0b, then the respective GLNVM_ULD.CONF_*_DONE is set to 1b by hardware.
 - After EMP firmware loads, during its init sequence, it sets the GLNVM_EMPLD.EMP_*_DONE bit (for each of CORER and GLOBLR domains). As a result, hardware sets the GLNVM_ULD.CONF_*_DONE bit to 1b.

4.2.1.4 Firmware Initialization

Once EMP firmware loads, the following X710-TM4 features are enabled:

- The external Ethernet link is active.
- Default internal switching components have been configured.
- Communication with the BMC is possible if supported by the X710-TM4 and the system (optional).
- OEM-specific manageability agents are active and responding to commands from the Ethernet fabric (optional).



- Admin queues for all enabled PCI functions are ready for commands. The EMP responds to each function's Get Version AQ command to indicate that it is safe for software to start using the X710-TM4 for software device driver initialization (see [Section 7.10.3](#)).
- HMC default profile has been configured.

Once EMP firmware is up and running, the X710-TM4 can be configured via its management interfaces and certain device capabilities are then enabled or disabled (such as soft SKUing).

4.2.1.5 MAC Address Initialization

Two sets of station LAN MAC addresses are supported:

- A station MAC address per PF. These addresses are used by the internal switch for L2 filtering of Rx packets, by the Rx classification filters, and for WoL purposes. The addresses are loaded from the NVM.
 - The station MAC addresses for WoL are loaded by the EMP into the PRTPM_SAL and PRTPM_SAH registers (see [Section 5.4.4](#))
 - A PF address is loaded per port into PRTPM_SAL[0,Port] and PRTPM_SAH[0,Port] registers.
- A station MAC address per Ethernet port. These addresses are used for link-level functionality such as flow control frames.
 - The PRTGL_SAL and PRTGL_SAH registers contain these addresses for the four 10 Gb/s MACs.

The following table lists how each address can be set or read and where it is stored:

MAC Address Type	Where Stored	How Reported	How Modified
LAN MAC address - factory	NVM PF MAC address section		N/A
LAN MAC address - current	NVM PF MAC address section alternate RAM	Manage MAC address read (LAN address valid)	Write alternate AQ command: Alternate LAN MAC address (LS) Alternate LAN MAC address (MS) or NC-SI Set Address OEM command or manage MAC address write (update LAA only)
Port MAC address	GLOBR registers auto-load module (PRTGL_SAL/H)	Manage MAC address read (port address valid)	Manage MAC address write AQ command (update port address) Note: Every function within a port is allowed to use this command – the last command takes precedence.
LLDP MAC address	Use the port MAC address for SA, chassis and port IDs.		
WoL MAC address	NVM PF MAC address section alternate RAM (PRTPM_SAL/H), PRTPM_SAL/H	Manage MAC address read (WoL address valid)	Manage MAC address write AQ command (update LAA and WoL address)
LAA MAC address	Local firmware variable	Manage MAC address read (LAN address valid)	Manage MAC address write AQ command (update LAA address) Note: Changing the LAA only does not impact the address used to detect magic packets.

The following rules apply:

- The *PF_NUM* field in the PRTPM_SAH register is deducted from the PF issuing the command
- The *AV* field in the PRTPM_SAH register is set by the EMP when writing a MAC address

The per-PF MAC address used by the internal switch is managed via the Remove MAC, VLAN pair and Add MAC, VLAN pair commands.



4.2.1.5.1 Manage MAC Address Read Command

This command is used by the PF driver to read the per-PF station MAC address.
Indirect command.

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0107	Command opcode.
Datalen	4-5	0x00	0x18 (the response buffer size).
Return Value/ VFID	6-7	0x00	Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by the firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque Value. Copied by the firmware into the completion of this command
Command Flags	16-17	Reserved	Zeroed by the software device driver.
Reserved	18-23	0x0	Reserved.
Data Address High	24-27	Buff Addr	High bits of return buffer address.
Data Address Low	28-31	Buff Addr	Low bits of return buffer address.

4.2.1.5.2 Manage MAC Address Read Response

A firmware acknowledge to the Manage MAC Address Read command.
Indirect Response.

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0107	Command Opcode.
Datalen	4-5	0x00	0x18 (the response buffer size).
Return Value/ VFID	6-7	Return Value	Return Value. Firmware supplies in the <i>Return Value</i> field indication on the completion of the Manage LAA command. 0x0 = No error. Others = Error detected in the command.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware from the manage MAC Address Read command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware from the manage MAC Address Read command.
Command Flags	16.0-16.3	Reserved	Zeroed by the EMP.
	16.4	LAN Address Valid	
	16.6	Port Address Valid	



Name	Bytes.Bits	Value	Remarks
	16.7	WOL Address Valid	
	17.0	MC_MAG_EN	If set, multicast magic packets generate a WoL event (if WoL is enabled for this function).
	17.1	WoL_preserve_on_PFR	Set to preserve WoL MAC on PFR (to be set by software before D0->Dr/D3 transition, cleared by software after returning to D0 state). 0b by default after POR/EMPR. Relevant only if WOL MAC was set.
	17.2-7	Reserved	
Reserved	18-23	0x0	Reserved.
Data Address High	24-27	Buff Addr	High bits of return buffer address.
Data Address Low	28-31	Buff Addr	Low bits of return buffer address.

Note: All MAC addresses are in big Endian order.

Address	Offset	Description
PF LAN SA	0-5	Current device value of the PF LAN MAC address. Validated by <i>LAN Address Valid</i> flag. This address is returned from LAA address if valid; otherwise, from alternate RAM if valid; otherwise, from the NVM if valid.
Port SA	12-17	Current device value of the Port MAC address. Validated by the <i>Port Address Valid</i> flag.
PF WoL SA	18-23	Current device value of the PF WoL MAC address. Validated by the <i>WoL Address Valid</i> flag.

4.2.1.5.3 Manage MAC Addresses Write Command

This command is used by the PF driver to write the per-PF station MAC address.

Direct command.

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0108	Command opcode. 0x0107 is used for reads; 0x0108 is used for writes.
Datalen	4-5	0x00	Must be 0x0, value is ignored.
Return Value/VFID	6-7	0x00	Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque Value. Copied by the firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by the firmware into the completion of this command.



Name	Bytes.Bits	Value	Remarks
Command Flags	16	Reserved	Zeroed by the software device driver.
	17.0	MC_MAG_EN	Used to set the MC_MAG_EN bit.
	17.1	LAA_WOL_PRESERVE	LAA WoL preserve on PFR.
	17.5 - 17.2	Reserved	Reserved.
	17.7:6	Write Type	00b = Update LAA only. 01b = Update LAA and WOL address. 10b = Update port address. 11b = Reserved.
SAH	18-19	See Remarks	High 16 bits of the MAC address (big endian order). For example, if MAC address = 11:22:33:44:55:66 then SAH = 0x1122.
SAL	20-23	See remarks	Low 32 bits of the MAC address (big endian order). For example, if MAC address = 11:22:33:44:55:66 then SAL = 0x33445566.
Data Address High	24-27	Buff Addr	High bits of return buffer address.
Data Address Low	28-31	Buff Addr	Low bits of return buffer address.

4.2.1.5.4 Manage MAC Address Write Response

A firmware acknowledge to the Manage MAC address write command.

Direct response.

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0108	Command opcode.
Datalen	4-5	0x00	Must be 0x0, value is ignored.
Return Value/ VFID	6-7	Return Value	Return value. Firmware supplies in the <i>Return Value</i> field indication on the completion of the Manage MAC Address write command. 0x0 = No error. Others = Error detected in the command.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware from the manage MAC Address Write command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware from the manage MAC Address Write command.



Name	Bytes.Bits	Value	Remarks
Command Flags	16	Reserved	Zeroed by the EMP.
	17.0	MC_MAG_EN	Used to set the <i>MC_MAG_EN</i> bit.
	17.1	LAA_WOL_PR ESERVE	LAA WoL preserve on PFR. Relevant only if WOL MAC was set.
	17.2	Valid	Port number in byte 16 is valid.
	17.5 -17.3	Reserved	
	17.7:6	Write Type	00b = Update LAA only. 01b = Update LAA and WOL address. 10b = Update port address. 11b = Reserved.
Reserved	18-23	0x0	Reserved.
Data Address High	24-27	Buff Addr	High bits of return buffer address.
Data Address Low	28-31	Buff Addr	Low bits of return buffer address.

4.2.1.6 Power-on Device State

This section describes the specific setting of each of the X710-TM4’s components required for pre-boot operation. It describes the information loaded from the NVM (per component) and the state attained by each.

- **Manageability** — System management functionality, if enabled, is fully operational by the end of the EMP firmware initialization sub-stage. Some configuration is loaded from the NVM during the power on stage. Capabilities might include the following:
 - Sideband interfaces
 - Pass-through manageability (including packet filtering)
 - Preparation for OS-to-BMC traffic
 - Preparation for MCTP over PCIe operation
- **Internal MAC and PHY** — If either system management or APM WoL are enabled, then enabled LAN ports are brought up by firmware once EMP firmware initialization completes. Else, enabled LAN ports are brought up following PE_RST_N de-assertion.
 - See [Section 3.2](#) for more details.
- **Admin Queue (AQ)** — A queue (Tx and Rx pair) is activated per enabled PCI function. For example, an AQ is needed for BIOS to change the switch or scheduler configuration.
 - See [Section 7.10.3](#) for more detail.
- **Internal Switch** — The internal switch is configured from the NVM for basic switching capabilities to the EMP and the enabled PCI functions. First, the switch programmable logic is loaded, followed by configuration of a basic switch topology. The topology is for basic L2 functionality.
 - See [Section 7.4.9.4.2](#) for more detail.
- **DCB** — The NVM loads the LLDP and DCBx setting into the X710-TM4. Once EMP firmware initializes and link is up, firmware engages in DCBx negotiation. Firmware then makes the appropriate changes in the X710-TM4’s configuration based on the outcome of the DCBx protocol. At this stage, DCB capabilities (TCs, ETS, PFC) might be enabled.
- **Tx Scheduler** — As the switch VSIs are enabled, firmware allocates Tx scheduler queue sets per each PF VSI based on the NVM configuration. Each queue set is configured to default behavior. Firmware also generates the required handles to enable system software to update the



configuration of each queue set. If DCBx protocol runs, the outcome of the protocol exchange might translate into changes in scheduler configuration.

- See [Section 7.8.4.1](#) for more detail.
- **Host Memory Cache (HMC)** — NVM settings supply the initial HMC configuration using a simple set of rules that enable equal distribution of HMC resources to all PFs that are connected to external Ethernet ports.
- **Power Management** — Some power management capabilities are supported pre-boot or during BIOS initialization.
 - The X710-TM4 might be enabled for APM wake during power up. See [Section 5.4.1](#) for more detail on how APM wake is configured.
 - EEE might be enabled once EMP firmware is initialized. See [Section 5.3.1.4](#) for more details.
- **LAN** — LAN queues are available for network boot in the BIOS initialization phase. Some LAN configuration is loaded from the NVM during power on, including partitioning of the LAN queues among PFs.

4.3 BIOS Init

This section describes how BIOS code might update the X710-TM4's configuration and the capabilities for network boot. The following sections are provided:

- [Section 4.3.2](#) describes how BIOS code might change the X710-TM4's configuration on each boot
- [Section 4.3.3](#) describes some aspects of supporting network boot
- [Section 4.3.4](#) describes the specific setting of each of the X710-TM4's components required for the BIOS Init stage

4.3.1 Initial State

The state of the X710-TM4 when BIOS begins to access it is described in [Section 4.3.4](#).

BIOS code must check that the EMP completed device initialization. This is done through the Get Version AQ command described in [Section 7.10.10.1](#).

4.3.2 Non-persistent Configuration

During power up, the X710-TM4 is factory configured from the NVM. However, the factory configuration might be overridden in two possible ways:

- **Persistent Configuration** — System tools (such as software agents and SMCLP commands) might write into the NVM and change the factory defaults. It is also possible to add alternate values into the NVM so that the factory defaults are preserved and might be restored at a later time.
- **Non-persistent Configuration** — An on-die Alternate RAM structure is provided to store configuration values that are not supposed to be maintained between cold resets. The information in this structure is retained during all resets other than a cold reset of the device.



The alternate structure is 8 KB, partitioned into 32-bit entries. Accessing the structure is done by addressing 32-bit entries. An address is therefore 11 bits (2 K Dwords), where address 0x000 points to the beginning of the 8 KB memory.

During a cold reset sequence, the alternate structure might be written by either an external system management agent (via the device management interfaces) or by BIOS level code (like SMASH/CLP commands).

The information is loaded into the X710-TM4 following the appropriate resets (refer to the text that follows), overriding the respective configuration loaded previously from the NVM. This section describes the details of how the alternate structure is handled.

The following mechanisms are provided:

- The alternate structure is committed to the device by a Done Alternate Write command in the following cases:
 - As part of the SMASH/CLP programming
 - Legacy BIOS—once the Init function is called for the 1st time in the flow
 - UEFI— once for each enabled LAN port
 - As part of executing the following resets: PE_RST_N, PCIR, EMPR, global reset, core reset. In other words, the contents of the alternate structure are only reset on cold resets.
- Option ROM code and UEFI drivers communicate with the X710-TM4 via a set of AQ commands:
 - The Read Alternate - Direct and Read Alternate - Indirect commands read from the alternate structure.
 - The Write Alternate - Direct and Write Alternate - Indirect commands write into the alternate structure.
 - The Done Alternate Write command indicates to the X710-TM4 that the CLP strings phase is done (for the entire device in Legacy BIOS mode and per LAN port in UEFI mode).

4.3.2.1 Alternate RAM Structure

The alternate structure is 8 KB, partitioned into 32-bit entries. Accessing the structure is done by addressing 32-bit entries. An address is therefore 11-bits (2 K Dwords), where address 0x000 points to the beginning of the 8 KB memory.

The structure is partitioned as follows:

Section	Address (DW)	Size (DW)	Content
Per PF	0 - 1023	1024 - 64 per PF	16 per-PF sections, one per PF. These sections are described in Section 4.3.2.1.1 and can be accessed by each PF.
EMP	1024 - 1279	256	Reserved for EMP use, including error logging. Can be written and read only by the EMP. In debug mode, this section can be read by the PFs.
Boot Configuration	1280 - 2047	768	A boot configuration section used for SAN boot configuration. This section is for software use only and the X710-TM4 is not aware of its internal content. All PFs can access this section (no enforcement by the X710-TM4).



4.3.2.1.1 Per PF Sections

Table 4-5. Per PF Alt RAM Content

Scope	Address (DW)	Contents	Used By	PCIe ALT Module in Shadow RAM	GLOBR Registers Auto load	CORER Registers Auto load
PF	0	Current LAN MAC Address (LS)	HW, SW			X
PF	1	Current LAN MAC Address (MS)	HW, SW			X
PF	18	SRIOV	HW	X		X
Reserved	6-63	Reserved	N/A			

Each entry is associated with one or more NVM modules and is loaded into the X710-TM4 following reset events that load these modules. Loading into the PCIe units is an exception. Parameters loaded into PCIe units are written by the EMP into the PCIe ALT auto-load module in shadow RAM. This is done as part of the BIOS initialization flow. On later resets, the X710-TM4 automatically loads the module into the PCIe units, saving the need for the EMP to intervene.

The PCIe ALT auto-load module is a regular hardware type 1 auto-load section as described in [Section 6.1.3.1](#). The content is dynamically created according to the registers that requires an auto-load value different than the default value or the value loaded from the NVM.

4.3.2.1.1.1 Current LAN MAC Address (offset 0x0, 0x1)

Lower Dword:

Field	Bit(s)	Description
MAC Address	31:0	MAC Address. Contains the LS 32-bit of the address.

Upper Dword:

Field	Bit(s)	Description
MAC Address	15:0	MAC Address. Contains the MS 16-bit of the address.
Reserved	30:16	Reserved.
Valid	31	Valid Bit. 0b = The MAC address two Dwords are invalid and should be skipped. 1b = The MAC address two Dwords are valid and should be processed.

Upper Dword:



Actions taken on a change in this entry:

- When valid, it overrides the MAC address loaded from the NVM
- In SFP mode:
 - Overrides the per-PF Station MAC Address (used by the internal switch)
 - Loaded into the PRTPM_SAL and PRTPM_SAH registers, to be used for WoL

4.3.2.1.1.2 SRIOV (Offset 0x12)

Field	Bit(s)	Description
SRIOV VFs	6:0	Number of SR_IOV VFs requested for the function. Can be any number between 0 and 128-1.
Reserved	28:7	Reserved.
Enable	29	SR-IOV is enabled (global configuration - valid only in function 0).
Reserved	30	Reserved
Valid	31	Valid Bit. 0b = Dword is invalid and should be skipped. 1b = Entry is valid and should be processed.

SFP mode uses the following flow:

- If the *Enable* bit is cleared (in function 0):
 - Write to PCIe alternate autoload section, *GLPCI_CAPSUP.SRIOV_EN* = 0b.
- If the *Enable* bit is set (in function 0):
 - Write to PCIe Alternate autoload section, *GLPCI_CAPSUP.SRIOV_EN* = 1b and *GLPCI_CAPSUP.ARI_EN* = 1b.

The EMP then calculates the number of interrupt vectors requested per VF through the *GLPCI_CNF2.MSI_X_VF_N* registers (done per enabled VF):

- If the value in the *GLPCI_CNF2.MSI_X_VF_N* field equals the maximum number of vectors (as listed in [Table 7-126](#), using the original number of VFs), then the EMP updates *MSI_X_VF_N* to equal the number calculated per [Table 7-126](#), this time with the updated number of enabled VFs.

Note: This is the case where the device requests the maximum number of possible vectors.

- Else, *MSI_X_VF_N* is not changed. This is the case where the device requests less than the maximum number of possible vectors.

Note: The EMP does not allocate MSI-X vectors to VFs. The PF should allocate them before SR-IOV is enabled.

- EMP updates the following registers in shadow RAM PCIe ALT Auto-load module.
 - *PF_VT_PFALLOC_PCIE*
 - *GLGEN_PCIFCNCNT*
 - *GLPCI_CNF2*
- EMP updates the following register fields (so that hardware is updated immediately):
 - *PF_VT_PFALLOC*
 - *GLGEN_PCIFCNCNT.PCIVFCNT*
 - *GLPCI_CNF2.MSI_X_VF_N*



- As part of each internal CORER flow, the EMP updates the following registers:
 - PF_VT_PFALLOC
 - GLGEN_PCIFCNCNT_INT
- The Discover Device/Function Capabilities command *Virtual Function* field reflects the change in allocations (see [Section 7.10.10.7](#)).
- Update the number of VSIs and MAC addresses allocated to each PF to be for each PF and one for each VF allocated to this PF. The new allocation should be stored in the SR PF allocation section.

Note: After receiving a Done Alternate Write command, firmware checks the total number of VFs allocated is not larger than 128, and if larger, returns an ENOSPC error.

4.3.2.2 AQ Commands

The [Table 4-6](#) lists the different AQ commands used to manage the alternate structure.

Table 4-6. List of AQ Commands for the Alternate Structure

Command	Opcode	Brief Description	Detailed Description
Write Alternate - Direct	0x0900	Write up to two parameters into the alternate structure.	Section 4.3.2.2.1
Write Alternate - Indirect	0x0901	Write a block of parameters into the alternate structure.	Section 4.3.2.2.2
Read Alternate - Direct	0x0902	Read up to two parameters from the alternate structure.	Section 4.3.2.2.3
Read Alternate - Indirect	0x0903	Read a block of parameters from the alternate structure.	Section 4.3.2.2.4
Done Alternate Write	0x0904	Indication that all CLP strings (for the entire X710-TM4 in legacy BIOS mode and per LAN Port in UEFI mode) have been sent to the X710-TM4.	Section 4.3.2.2.5
Clear Port Alternate	0x0906	Clear content of alternate RAM relevant to this port.	Section 4.3.2.2.6

4.3.2.2.1 Write Alternate - Direct

The Write Alternate - Direct command writes to the alternate structure up to two parameters.

Table 4-7. Write alternate - Direct Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0900	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7	0x00	N/A
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
First Parameter Address	16-19		Accessing the alternate structure is done by addressing 32-bit entries.



Table 4-7. Write alternate - Direct Command

Name	Bytes.Bits	Value	Remarks
First Parameter Data	20-23		
Second Parameter Address	24-27		Address should be within the range allocated to the function inside the alternate structure. Value of 0xFF..FF means only the first parameter is written. Accessing the alternate structure is done by addressing 32-bit entries.
Second Parameter Data	28-31		

Table 4-8. Completion for the Write Alternate - Direct Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0900	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7		Some comments on specific errors: 0x0 = No error. ENOMEM = Out of memory (access outside the alternate structure). EACCES = Permission denied (access to another PF's area).
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31		Might contain the values sent in the original command.

4.3.2.2 Write Alternate - Indirect

The Write Alternate - Indirect command writes a block of parameters to the alternate structure. The command defines the number of Dwords to be written and the starting address inside the alternate structure.

Table 4-9. Write Alternate - Indirect Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0901	Command opcode.
Datalen	4-5		Size of buffer accompanying the command (in bytes).
Return Value/ VFID	6-7	0x00	N/A
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Alternate Structure Address	16-19		Lowest address to be written into the alternate structure. Accessing the alternate structure is done by addressing 32-bit entries.
Alternate Structure Length	20-23		Number of Dwords to be written into the alternate structure.
Data Address High	24-27	Buff Addr	High bits of buffer address.
Data Address Low	28-31	Buff Addr	Low bits of buffer address.



The following completion is sent for the Write Alternate - Indirect command:

Table 4-10. Completion for the Write Alternate - Indirect Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0901	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7		Some comments on specific errors: 0x0 = no error. ENOMEM = Out of memory (access outside the alternate structure). EACCES = Permission denied (access to another PF's area).
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31		

4.3.2.2.3 Read Alternate - Direct

The Read Alternate - Direct command reads from the alternate structure up to two parameters.

Table 4-11. Read Alternate - Direct Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0902	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7	0x00	N/A
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
First Parameter Address	16-19		Address should be within the range allocated to the function inside the alternate structure. Accessing the alternate structure is done by addressing 32-bit entries.
Reserved	20-23	0x00	
Second Parameter Address	24-27		Address should be within the range allocated to the function inside the alternate structure. Value of 0xFF..FF means only the first parameter is read. Accessing the alternate structure is done by addressing 32-bit entries.
Reserved	28-31	0x00	

The following completion is sent for the Read Alternate - Direct command:

Table 4-12. Completion for the Read Alternate - Direct Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0902	Command opcode.
Datalen	4-5	0x00	N/A



Table 4-12. Completion for the Read Alternate - Direct Command

Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Some comments on specific errors: 0x0 = no error. ENOMEM = Out of memory (access outside the alternate structure). EACCESS = Permission denied (access to another PF's area).
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
First Parameter Address	16-19		Copied from command.
First Parameter Data	20-23		Data read.
Second Parameter Address	24-27		Copied from command. Value of 0xFF.FF means only the first parameter is read.
Second Parameter Data	28-31		Data read.

4.3.2.2.4 Read Alternate - Indirect

The Read Alternate - Indirect command reads a block of parameters to the alternate structure. The command defines the number of Dwords to be read and the starting address inside the alternate structure.

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0903	Command opcode.
Datalen	4-5		Size of buffer accompanying the command (in bytes).
Return Value/ VFID	6-7	0x00	N/A
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Alternate Structure Address	16-19		Lowest address to be read from the alternate structure. Accessing the alternate structure is done by addressing 32-bit entries.
Alternate Structure Length	20-23		Number of Dwords to be read from the alternate structure.
Data Address High	24-27	Buff Addr	High bits of buffer address.
Data Address Low	28-31	Buff Addr	Low bits of buffer address.

The following completion is sent for the Read Alternate - Indirect command:

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0903	Command opcode.
Datalen	4-5	0x00	Actual length of data returned by the command.



Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Some comments on specific errors: 0x0 = no error. ENOMEM = Out of memory (access outside the alternate structure). EACCES = Permission denied (access to another PF's area).
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Alternate Structure Address	16-19		Lowest address read from the alternate structure.
Alternate Structure Length	20-23		Number of DWs read from the alternate structure.
Data Address High	24-27	Buff Addr	High bits of buffer address.
Data Address Low	28-31	Buff Addr	Low bits of buffer address.

4.3.2.2.5 Done Alternate Write

The Done Alternate Write command indicates to the X710-TM4 that the CLP strings have been sent to it:

- **Legacy BIOS Mode** — Sent once per device after all CLP strings have been sent to the X710-TM4 (for all LAN ports). Following the command, firmware loads the contents of the alternate structure into the device functional units.
- **UEFI Mode** — Sent once per each enabled LAN port. Once the command is received from all enabled ports, firmware loads the contents of the alternate structure into the X710-TM4's functional units.

After this command, software should trigger a shadow RAM dump using a dummy NVM update with *Last* flag set (bit 19.1).

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0904	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7	0x00	N/A
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
BIOS Mode	16.0	See remarks	0b = Legacy BIOS. 1b = UEFI.
Reserved	16.7 - 16.1	0x00	Reserved.
Reserved	17-31	0x00	Reserved.



The following completion is sent for the Done Alternate Write command:

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0904	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7		ENOSPC - more than 128 VFs are requested.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16.0	0b	Reserved.
Return Flags	16.1		Reset Needed. When set, indicates that software should do a global reset for the alternate RAM content to take effect.
Reserved	16.2-31	0x00	

4.3.2.2.6 Clear Port Alternate Write

The Clear Port Alternate command indicates to the X710-TM4 to clear the alternate sections. The port is inferred from the PF that sent the command.

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0906	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7	0x00	N/A
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31	0x00	

The following completion is sent for the Clear Port Alternate command:

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0905	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7		



Name	Bytes.Bits	Value	Remarks
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Reserved	16-31	0x00	

4.3.2.3 Example of a SMASH/CLP Flow - Legacy BIOS

The following pseudo code provides an example of how such a flow might be performed by legacy BIOS using SMASH/CLP commands. The following guidelines should be kept:

- If a certain PCI function is disabled via this mechanism, pre-boot software does not access any resource of that function (such as any CSR) once it sends the Done Alternate Write AQ command. The X710-TM4 confirms the command, resets, and disables the function.
- By the time the SMASH/CLP commands are executed and a function is disabled, there is no Tx/Rx activity in the X710-TM4 (since no queues have been initialized).
- All ports enabled in the NVM have the option ROM enabled (such as the lowest PCI function per port has an expansion ROM BAR).
- BIOS calls all CLP entry points for all functions before getting into the initialization phase.
- It is not guaranteed that a system reset is issued immediately following this sequence. For example, the configuration settings must take place even if such a reset is not issued.
- During the initialization phase, pre-boot software has to issue a global reset followed by a Start LLDP Agent AQ command to start the LLDP agent in firmware.

```
// Initial state:
// The device is configured from NVM
// The following flow runs per each device
BIOS does PCI enumeration and discovers functions with Option ROM enabled
For each LAN Port with Option ROM enabled // sent to the lowest number PCI function on the port
    BIOS loads the Option ROM into system RAM
    Set First-Init to true // When Init is later called, it's the 1st call to Init
    For each CLP string received by BIOS for a function within the port
        If the Option ROM for the port supports SMCLP entry point
            BIOS calls SMCLP entry point for port with CLP string
            SMCLP section in Option ROM processes the CLP string
            Option ROM keeps track of SMCLP status for the port
        End-If
    End-For
End-For
For each LAN port with Option ROM enabled
    BIOS calls INIT entry point for the port
```



```
End-For
SMCLP Entry Point
If CLP type is SET and action is "Return to default" // clear any pre-existing CLP configuration
    Send the "Clear Port Alternate" admin command to invalidate all Alternate Memory parameters
    for the port and its PFs
End-If
If CLP type is SET
    Generate "Write Alternate - Direct" admin command(s) to the device
    End-If
If CLP type is EXIT // Apply any configuration changes from previous commands that have not
    been applied
    Send appropriate admin commands with default values for the PF parameters not specified by the
    CLP string that have a default behavior in this mode, different than the hardware default.
End-If
End SMPCLP Entry Point
SMCLP INIT Entry point
If First-Init is true // the flow below should only be executed on the 1st call to Init
    If SMCLP status is false // means no CLP strings have been received for any functions for the
    device
        // Option ROM INIT configures the device in a standard operating mode (i.e. not an OEM
        specific mode)
        End-if
    If SMCLP status is true // means at least one function had CLP strings
        // This is the time to load the CLP parameters from the Alternate Structure into the
        device.
        Send "Done Alternate Write" command indicating end of CLP strings for the device
        End-if
        // Global Reset should be done only once per device
        If Firmware required a reset in the "Done Alternate Write" response, Issue Global Reset of the
        device
            Set First-Init to false
        End-if
    Continue with INIT code...
End SMCLP INIT Entry Point
```



4.3.2.4 Example of a SMASH/CLP Flow - UEFI

The following pseudo code provides an example of how such a flow might be performed by the UEFI drivers using SMASH/CLP commands. The following guidelines should be kept:

- If a certain PCI function is disabled via this mechanism, pre-boot software does not access any resource of that function (such as any CSR) once it sends the Done Alternate Write AQ command for that function. The X710-TM4 confirm the command, resets, and disables the function.
- By the time the SMASH/CLP commands are executed and a function is disabled, there is no Tx/Rx activity in the X710-TM4 (since no queues have been initialized).
- A UEFI driver is executed for each enumerated LAN port. For example, any port enabled in the NVM and not disabled by strapping.
- It is not guaranteed that a system reset is issued immediately following this sequence. For example, the configuration settings must take place even if such a reset is not issued.

```
// Initial state:
// The device is configured from NVM
// The following flow runs per each device
BIOS does PCI enumeration and discovers PCI functions
For each enabled LAN Port
    BIOS calls driver START entry point
    For each CLP string received by BIOS for a function within the port
        SMCLP section processes the CLP string
    End-For
    Driver sends a "Done Alternate Write" command indicating end of CLP strings for the port
    If Firmware required a reset in the "Done Alternate Write" response, Issue Global Reset of the
    device
End-For
SMCLP section in START
If CLP type is SET and action is "Return to default" // clear any pre-existing CLP configuration
    Send the "Clear Port Alternate" admin command to invalidate all Alternate Memory parameters for
    the port and its PFs
End-If
If CLP type is SET
    Generate "Write Alternate - Direct" admin command(s) to the device
End-If
If CLP type is EXIT // Apply any configuration changes from previous commands
    // that have not been applied
    Send appropriate admin commands with default values for the PF parameters not specified by the
    CLP strings that have a default behavior in this mode, different than the hardware default.
End-If
```



End SMPCLP section

4.3.2.5 Processing the Alternate Structure

As a result of each Done Alternate Write AQ command, the EMP loads the relevant alternate structure parameters into the X710-TM4 according to the following sequence:

- EMP loads any required parameters from the alternate structure into the hardware
- EMP ACKs the Done Alternate Write command
- Global reset is performed
 - Initiated by software
 - Reset is done only once and not per each instance of the Done Alternate Write AQ command
- Hardware is initialized, including loading of the relevant sections of the Alternate Structure into hardware

4.3.3 Network Boot

Each PF can be a boot function and can independently support PXE, iSCSI boot:

- Any functions can be a PXE boot function (up to 16 such functions)
- Up to four functions might be iSCSI boot functions

4.3.4 Device State

This section is limited to changes in the X710-TM4's state made in the BIOS initialization stage. For units not mentioned here, behavior is as described in [Section 4.2.1.6](#).

4.3.4.1 Switch / Tx Scheduler

Some parameters of the switch and Tx scheduler configuration might change by the contents of the alternate structure. Such changes take effect when the alternate structure is enabled.

4.3.4.2 LAN

Expansion ROM code might set LAN QPs for network boot. The sequence of setting a LAN QP is described in [Section 8.2.2](#).

4.3.4.3 Interrupts

Interrupts need to be set if used for admin queue or LAN queues operation. See [Section 7.5.1.1](#) for the exact flow.



4.4 Driver Load

4.4.1 Introduction

4.4.1.1 Driver Load (Non-virtualized)

As described earlier in this section, by the time the software device driver loads, the NVM configuration already completed and PCIe configuration has taken place. During a software device driver load, the following sequence of commands is typically issued in the X710-TM4 to initialize it for normal operation. The major initialization steps are:

1. Device driver probes the X710-TM4 for resource allocations.
2. Disable interrupts.
3. Initialize the admin queue - see [Section 7.10.3](#).
4. Initialize HMC - see [Section 7.9.3](#).
5. Initialize MAC/PHY - see [Section 3.2.1](#).
6. Initialize power management - see [Section 5.3.1.4 \(EEE\)](#), [Section 5.4.5 \(Wake Up\)](#).
7. Initialize DCB (including Rx-PB)
8. Initialize the switch and Tx scheduler - see [Section 7.4.9.4](#) and [Section 7.8.4.19](#).
9. Initialize statistics by reading the counter's initial values to serve as a baseline.
10. Initialize 1588.
11. Enable interrupts.

At this point, the X710-TM4 is ready to initialize VSIs and LAN flows. These are done dynamically during operation:

- Initialize VSI.
- Initialize LAN QP; including its interrupts (see [Section 8.2.2](#)) or configure filters (see [Section 7.1](#)).



4.4.1.2 Driver Load (SR-IOV)

The X710-TM4's approach for virtualized device drivers is to depend heavily on the PF device driver for management of chip resources. The software device driver models for newer virtualized operating systems are moving in a direction that require such functionality. Figure 4-10 shows the high-level initialization flow for virtualized device drivers that use a VF in a guest operating system.

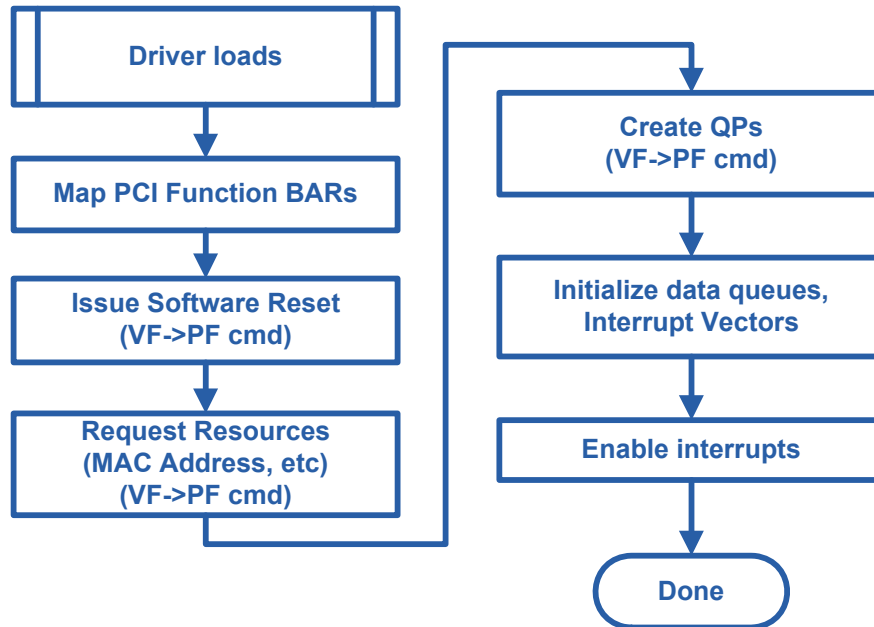


Figure 4-10. Virtualized Device Driver Initialization Flow

4.4.1.2.1 PF Initialization Details

In addition to the regular software device driver initialization flow described in Section 4.4.1.1, the PF device driver should apply the following steps to enable support for VMs.

1. After the operating system enables virtual bridging, the PF device driver should create a VEB or a VEPA switching element using the following flow:
 - a. Query the switch structure using the Get Switch Configuration command (Section 7.4.9.5.3.1) in order to get the SEID number to which this PF is connected. This SEID might be the port or an S-channel.
 - b. Create a VEB/VEPA using the Add VEB command (Section 7.4.9.5.7.1). The control port can be either the original VSI of the PF or a dedicated VSI.
 - c. Connect the switch in place of the current VSI using the Insert Element command (Section 7.4.9.5.7.1).
 - d. Connect default and mirror ports as needed using the Add VSI and Connect Elements commands (Section 7.4.9.5.5.1 and Section 7.4.9.5.5.3).
 - e. Define mirroring rules using the Mirroring Rules commands (Section 7.4.9.5.9.13).
 - f. Activate malicious driver protection through the GL_MDCK_RX GL_MDCK_TDAT and GL_MDCK_TCMD registers.



2. When the VMM requests that a virtual port be created, the PF driver should:
 - a. Create a set of VSIs according to the flow described in the text that follows. The exact flow depends on the VMM-to-PF API.
 - b. Define the bandwidth allocated to the VSI and each of its TCs using the Scheduler Configuration commands (Section 7.8.4).
 - c. If the virtual port is a VF, allow VF access to the network by clearing the associated bit in the GL_VIRT_VFLRE register.

4.4.1.2.2 VF Initialization Details

This section describes the flow used to initialize a VF. It refers to various stages shown in Figure 4-10. Only stages involving the hardware are detailed.

4.4.1.2.2.1 Software Reset

A VF software reset can be asserted only by the PF using the *VPGEN_VFRTRIG.VFSWR* field. Following a VF software reset, the VF should request re-initialization of the queues from the PF.

Following the reset, the PF should preform the clean-up steps described in Section 4.1.2.5.

4.4.1.2.2.2 Request Resources and Create Initialize Data Queues

On top of the resources statically allocated to a VF (interrupts, RSS table, etc.), a VF might have a set of VSIs. Each VSI contains objects such as:

- User priorities (transmit queue groups).
- Queue pairs
- Queuing filters

VSIs can be requested for LAN.

The resources should be requested in the following order (operations with the same stage number can be done in any order):

Step	Resource Requested	PF Action	Notes
1	VSI and UPs	Allocate VSIs according to allocation policies using the Add VSI AQ command (Section 7.4.9.5.5.1).	The PF should activate the security features in the VSI (anti spoof, port based VLAN) according to the VF settings.
2	Forwarding table entries (MAC and VLANs)	Add the requested MAC and VLAN and MAC, VLAN pairs using the forwarding table configuration commands (Section 7.4.9.5.9).	
3	Data queue pairs	Create the requested queue contexts as described in Section 8.4.3.1 and Section 8.3.3.1.	



4.5 Device/Port/Function Configuration

4.5.1 General

The X710-TM4 provides several mechanisms to configure which PCI functions and Ethernet ports are exposed.

The functions and ports can be hidden or exposed:

- Through NVM configuration
- Bases on power management policy
- Through strapping pins

4.5.1.1 Port-to-Function Mapping

The PFGEN_PORTNUM.PORT_NUM per-PF register field (loaded from the NVM) associates each PF with a port.

4.5.2 Disable Through Strapping Pins

For a LOM design, it might be desirable for the system to provide BIOS setup capability for selectively enabling or disabling the X710-TM4 PCI devices, PCI functions, or external Ethernet ports and the associated PCI function. It enables end users more control over system resource management and avoids conflicts with add-in NIC solutions. The X710-TM4 provides support for selectively enabling or disabling one or more LAN PCI device(s) in the system.

The X710-TM4 provides options to disable its external Ethernet ports and/or PCI functions:

- A strapping pin (DEV_DIS_N) is sampled on LAN_PWR_GOOD and PCIe resets to disable Ethernet ports. The specific port(s) to be disabled is determined from NVM. Additional NVM configuration is provided to determine which PCI functions are disabled at the same time. The expected usage is to disable the PCI functions associated with the disabled ports.
- A strapping pin (PCI_DIS_N) is sampled on LAN_PWR_GOOD and PCIe resets to disable PCI functions. The specific functions that are disabled are determined from the NVM.

Some guidelines on usage of these straps:

- During power up, the PCI_DIS_N and DEV_DIS_N pins are ignored until the NVM is read. From that point, the X710-TM4 might disable all PCI functions if either PCI_DIS_N or DEV_DIS_N is asserted.
- De-assertion of the PCI_DIS_N or DEV_DIS_N pins requires the system to issue a power on reset/ LAN_PWR_GOOD/PE_RST_N/in-band reset to the X710-TM4 in order to re-enable any disabled PCI functions or external Ethernet ports.
- The PCI_DIS_N and DEV_DIS_N pins should maintain their state during system reset and system sleep states. It should also insure the proper default value on system power up. For example, one could use a GPIO pin that defaults to 1b (enable) and is on system suspend power (such that it maintains its state in S0-S5 ACPI states).



4.5.3 Port and Device Disable

The following mechanisms are provided to enable and disable Ethernet ports:

- NVM configuration:
 - Ports are enabled or disabled from the NVM. The PRTGEN_CNF.PORT_DIS per-port register bit (loaded from the NVM) disables external ports other than port 0 (PORT_DIS for port 0 is hard wired to always enabled).
 - When cleared, the respective port is enabled.
 - The hardware default value is for port 0 to be enabled and other ports to be disabled.
- Power management:
 - If the Ethernet ports are not required in Dr state (such as if manageability is disabled and APME WoL is disabled as well), then all ports are disabled during Dr state. See [Section 5.2.3.4](#).
- Strapping (DEV_DIS_N):
 - When the DEV_DIS_N pin is asserted low, the PRTGEN_CNF.ALLOW_PORT_DIS per-port bits (loaded from the NVM) determine if the port is disabled.
 - The hardware default value for these bits is 0x0 (do not disable).
 - If a bit is set to 0b, DEV_DIS_N has no effect on the port.

Note: If a port is required for manageability purposes, it should not be disabled by the mechanisms previously described.

The result of the previous mechanisms is reflected in the PRTGEN_STATUS.PORT_VALID bit (per port), which denotes if the port is enabled. A bit is cleared (port disabled) if at least one of the previous mechanisms disables the port. The port is then powered down (including MAC, PCS, PHY).

If all ports are disabled, the X710-TM4 is put in a power-down, reset state. Specifically, the X710-TM4 does not respond to PCI configuration cycles, the PCIe link is in L3 state, and Ethernet ports are in power down. All PCI functions must be disabled as well via the mechanisms previously described (such as the proper NVM configuration must be set to disable for PFs).

4.5.3.1 Dynamic Port Shutdown

Another related, per-port status bit is the PRTGEN_STATUS.PORT_ACTIVE indication. This is a dynamic state indicating that the port is temporarily inactive and is powered down. When the port is inactive, re-activating it, does not require any reset, and the related PCIe function is still active.

A port is active (PRTGEN_STATUS.PORT_ACTIVE = 1b) based on the following rules:

1. PRTGEN_STATUS.PORT_VALID = 1b, as previously described and the X710-TM4 is in D0 state (PMCSR.PowerState = D0).
 - If PRTPM_GC.EMP_LINK_ON is set to 1b (such as the interface is being used for manageability) then the link is kept on. Specifically, hardware ignores disabling the link using the PRTGEN_CNF.ACTIVATE_PORT_LINK.



- Else,
 - In systems where the application needs to prevent any traffic on link before the device driver is loaded, the PRTGEN_CNF2.ACTIVATE_PORT_LINK is loaded from the NVM with the value of 0b (disable)
 - Once the software device driver loads, it uses the Set Link and Restart AN with the command bit 2 set (Enable Link). Following this, the EMP enables the link by writing the value of 1b to the PRTGEN_CNF2.ACTIVATE_PORT_LINK and starts the link bring-up sequence.
 - If the software device driver is about to be removed or disabled, then before going down the software device driver might disable the link by the Set link and Restart AN with the command bit 2 cleared (Disable Link). Following this, firmware disables the link by writing the value of 0b to the PRTGEN_CNF2.ACTIVATE_PORT_LINK.
- 2. PRTGEN_STATUS.PORT_VALID = 1b, as previously described, and the X710-TM4 is in Dr state (PMCSR.PowerState = D3).
 - If port is enabled for wake-up, follow the description in [Section 5.4](#).
 - Else, behavior is defined by the EMP_LINK_ON bit:
 - The hardware default value for PRTPM_GC.EMP_LINK_ON is 0b (link should be down as it is not required for EMP functionality).
 - MC might enable manageability by sending the appropriate command (see [Section 9.6.3.1](#)). As a result, EMP transitions to pass-through manageability-on state, enables the respective port (if disabled) by setting the PRTPM_GC.EMP_LINK_ON bit for the port, and starts the link bring-up sequence.
 - If the channel is disabled by the MC (by sending the appropriate command (see [Section 9.6.3.1](#)), and EMP has no other needs for the LAN port in Dr state (like proxy), then the EMP reverts the PRTPM_GC.EMP_LINK_ON bit for the port to the value of EMP_LINK_ON bit in the NVM manageability module or is set to zero (in case of the Disable Channel command with the *ALD* bit set).

4.5.4 Function Disable

The following mechanisms are provided to enable and disable PCI functions.

- NVM configuration:
 - PCI functions are enabled or disabled from the NVM. The PFPCI_FUNC.FUNC_DIS per-PF bit (loaded from the NVM) disables PCI functions (other than function 0. FUNC_DIS for function 0 is hardwired to enabled).
 - When cleared, the respective function is enabled.
 - The hardware default value is for function 0 to be enabled and other functions to be disabled.
- Strapping through PCI_DIS_N:
 - When the PCI_DIS_N pin is asserted low, the PFPCI_FUNC.ALLOW_FUNC_DIS per-PF register bit (loaded from the NVM) determines if the function is disabled.
 - If this bit is set to 0b, PCI_DIS_N has no effect on the PCI function.
 - The hardware default value is 0b (keep enabled).
- Strapping through DEV_DIS_N:
 - When the DEV_DIS_N pin is asserted low, the PFPCI_FUNC.DIS_FUNC_ON_PORT_DIS per-PF bit (loaded from the NVM) determines if the function is disabled.
 - If this bit is set to 0b, DEV_DIS_N has no effect on the PCI function.
 - The hardware default value is 0 (do not disable).



- Soft SKU commands:
 - The soft SKU commands are executed either before or after PE_RST_N was de-asserted and in any case, before the first access by BIOS to the X710-TM4 (such as before the first PCIe configuration cycle to the X710-TM4).
 - When a LAN port is enabled or disabled via a Soft SKU command, the EMP identifies (through the PFGEN_PORTNUM registers) the PCI functions associated with the port.
 - EMP then updates the configuration of the functions enabled or disabled by writing to the PFPCI_FUNC.FUNC_DIS register bits.
 - PFPCI_FUNC.FUNC_DIS for function 0 is hardwired to enabled, so is not affected by this mechanism.
- Another configuration is:
 - The GLGEN_PCIFCNCNT register reflects the number of enabled PFs as loaded from the NVM (determined only by PFPCI_FUNC.FUNC_DIS). The value is loaded from the NVM.

The result of the previous mechanisms is reflected in the PFPCI_STATUS1.FUNC_VALID bit (per PF), which denotes if the function is enabled. A bit is cleared (function disabled) if at least one of the previous mechanisms disables the function.

When a function is disabled:

- It does not respond to PCI configuration cycles (unless specified otherwise). Effectively, the function becomes invisible to the system.
- The *PME_En* bit is cleared to avoid issuing PME.

The Ethernet ports associated with disabled PCI functions are still available for manageability purposes.

4.5.5 Event Flow for Enable/Disable Ports and PCI Functions

This section describes the expected flow to disable or enable PCI functions or Ethernet ports. Following a power-on reset/LAN_PWR_GOOD/PE_RST_N/in-band reset, the DEV_DIS_N and PCI_DIS_N signals should be driven high (or left unconnected) for normal operation.

The following example assumes that PCI functions and/or Ethernet ports are being disabled during the BIOS initialization phase:

1. Following a power-up sequence, the DEV_DIS_N and PCI_DIS_N signals are driven high.
2. The PCIe link is established following PE_RST_N.
3. BIOS goes through PCI bus enumeration.
4. BIOS recognizes that the PCI functions in the X710-TM4 should be disabled.
5. The BIOS drives the DEV_DIS_N or PCI_DIS_N signal to a low level.
6. The BIOS asserts PCIe reset, either in-band or via PE_RST_N.
7. As a result, the X710-TM4 samples the DEV_DIS_N and PCI_DIS_N signals and disables the PCI functions and/or external Ethernet ports.
8. BIOS might do device enumeration a second time (the disabled PCI functions are invisible or changed to dummy function).
9. Proceed with normal operation.
10. Re-enable could be done by driving the DEV_DIS_N and PCI_DIS_N signals high and re-issuing a power-on reset/LAN_PWR_GOOD/PE_RST_N/in-band reset.



4.5.5.1 Multi-function Advertisement

If all but one of the PCI functions are disabled, the X710-TM4 is no longer a multi-function device. The X710-TM4 normally reports a 0x80 in the PCI configuration header field header type, indicating multi-function capability. However, if only a single LAN is enabled, the X710-TM4 reports a 0x0 in this field to signify single-function capability.

4.5.5.2 Legacy Interrupts Use

Each NVM PF configuration module specifies the interrupt line used for each PCI function. When more than one PCI function is enabled, the X710-TM4 uses the INTA# to INTD# interrupts for interrupt reporting. The specific interrupt pin used is reported in the *PCI Configuration Header Interrupt Pin* field associated with each PCI function.

However, if only one PCI function is enabled, then the INTA# must be used for this PCI function, regardless of the NVM configuration. Under these circumstances, the *Interrupt Pin* field of the PCI header always reports a value of 0x1, indicating INTA# usage.

4.5.5.3 Power Reporting

When more than one PCI function is enabled, the PCI power management register block has the capability of reporting a common power value. The common power value is reflected in the *Data* field of the PCI Power Management registers. The value reported as common power is specified via the *LAN Power Consumption* NVM word (word 0x22), and is reflected in the *Data* field each time the *Data_Select* field has a value of 0x8 (0x8 = common power value select).

When only one PCI function is enabled, the X710-TM4 appears as a single-function device, the common power value, if selected, reports 0x0 (undefined value), as common power is undefined for a single-function device.

4.6 Shared Resource Management

The X710-TM4 is a device with multiple external Ethernet ports and that supports multiple PCI functions. Several on-chip resources are shared between software device drivers that load the PCI functions exposed by the X710-TM4. Additionally, some resources for SR-IOV VFs are managed by the associated PFs. In general, the X710-TM4 minimizes the requirement for coordination between software device drivers running on different PCI PFs through a combination of resource partitioning and programming interface assistance for remaining resources that are shared.



The X710-TM4 handles shared resources using a number of allocation mechanisms and/or access control mechanisms. Table 4-13 lists the supported mechanisms.

Table 4-13. Supported Mechanism

Class	Description
Dedicated	This resource is associated with a particular X710-TM4 element and is always available without respect to any other configuration or setting. Examples of elements in this context could be a PCI function or external Ethernet port.
Administered	Administered resources can be re-assigned based on MC, BIOS settings, or other OEM specific mechanisms. These resources can be changed with a PCI reset but do not change dynamically after the PCI reset.
Profiles	Resources that are allocated via profiles are typically divided up among different entities based on some combination of other input. A profile might dictate that a resource is divided among the active PCI functions in a particular manner. Another possible usage of a profile is to divide resources based on both the number of PCI functions and the number of external Ethernet ports. The division of resources that are allocated by profiles happens between the time that a PCI reset occurs and might be impacted by an initial software device driver load, but resource allocation based on profiles require a PCI reset to change after the initial software device driver load.
Pool	Resources that are allocated from pools are typically allocated in a fashion that guarantees no resource starvation to an individual consumer of a resource but enables flexible allocation of remaining resources to any consumer that requires additional resources beyond the minimum. Pools are used for resources that do not need to have a maximum number of resources allocated to all consumers at the same time and that the consumers are reasonably able to fail an allocation.
Service	Resources that are possibly allocated on a per-device or per-port basis but are accessed by a software service that has visibility to multiple software device driver interfaces that do not need the X710-TM4 to provide an arbitration mechanism. In these cases each PCI function associated with a given resource provides equal access to the resource as other PCI functions associated with the resource. This enables the software service to have the flexibility to access such resources from any software device driver instance that it finds available and to switch between software device driver instances as the software device drivers are stopped and started. Resources that need to be accessed directly by a software device driver without support from an overlaying software service cannot be handled in this fashion.
Arbitrated	Resources that are allocated on a per-device or per-port basis need an access control mechanism that enables multiple consumers of the resource to operate in an independent manner. The X710-TM4 does this by either providing a firmware interface to access the resource where firmware handles the requests one at a time or by other hardware based arbitration scheme.

Authentication enable controls the application of the authentication flow as part of the ROM boot flow. If not set, then a firmware that is part of a non-authenticated image is allowed to run.



5.0 Power Management

This section defines how power management is implemented in the X710-TM4.

5.1 Power Targets and Power Delivery

See [Section 13.4](#) for the current consumption and see [Section 13.3.1](#) for the power supply specification.

5.2 PCIe Power Management

5.2.1 Auxiliary Power Usage

Auxiliary power might be used for powering the X710-TM4 while the system is in a low power state (D3cold). The X710-TM4 uses the AUX_PWR pin as an indication that auxiliary power is available to it. The X710-TM4 uses this indication to advertise D3cold wake up support in the PMC.PME_Support field and to set the Aux_Power_Detected bit in the PCIe capability structure Device_Status_Register. The AUX_PWR pin is strapped during Power On Reset (POR). The amount of power required for the function, which includes the entire system is advertised in the Power Management Data register, which is loaded from the NVM.

If AUX power usage during D3cold is supported (as follows), all the X710-TM4's sticky bits preserve their values and only get reset by the power up reset (detection of power rising).

When AUX power is applied to the X710-TM4, the actual usage of AUX power during D3cold is controlled through these factors:

1. NVM loaded bits:
 - EMP_LINK_ON — Bit per port loaded from the NVM to the PRTPM_GC register that indicates if the relevant port link should be established for EMP functionality. The X710-TM4 provides the proper clocking to establish the required port link(s) at this state.
2. APME — Bit per function (loaded to PFPM_APM[PF]) indicates if the relevant PF associated to a port link should be established to enable APM WoL. The X710-TM4 provides the proper clocking to establish the required port link(s) at this state.
 - PME_En — The PME_En bit of the PMCSR PCI configuration space is used to determine if the X710-TM4 is allowed to consume AUX power for WoL.

The following pseudo code defines AUX power usage where the APME and PME_En bits refer to a logical OR over all the PFs attached to the port:

```
If (AUX_PWR = 1)
    if ((APME or EMP_LINK_ON or PME_En) = 1)
        AUX power is used to preserve link functionality
```

Else

Link functionality is not preserved during AUX power supply

5.2.2 PCIe Link Power Management

The PCIe link state follows the power management state of the device. Since the X710-TM4 incorporates multiple PCI functions, the device power management state is defined as the power management state of the most awake function:

- If any function is in a D0a state in ARI mode or either D0a or D0u in non-ARI mode, the PCIe link assumes the device is in the D0 state.

Else

- If in ARI mode, at least one of the functions is in the D3 state and the other functions are not in the D0a state, or if in non-ARI mode all of the functions are in the D3 state, the PCIe link assumes the device is in D3 state.

Else

- The device is in the Dr state (*PE_RST_N* is asserted to all functions).

The X710-TM4 supports all PCIe power management link states:

- L0 state is used in D0u and D0a states.
- The L0s state is used in D0a and D0u states each time link conditions apply.
- The L1 state is used in D0a and D0u states each time link conditions apply, as well as in the D3 state.
- The L2 state is used in the Dr state following a transition from a D3 state if PCI-PM PME is enabled.
- The L3 state is used in the Dr state following power up, on transition from D0a and also if PME is not enabled in other Dr transitions.

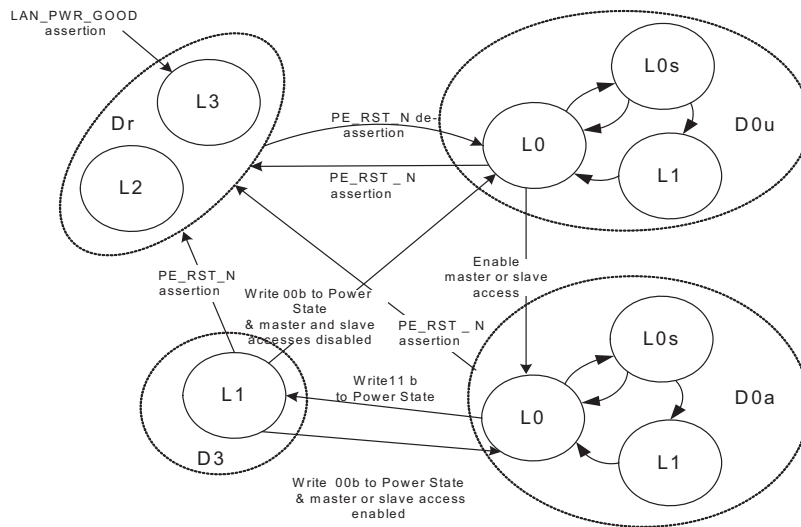


Figure 5-1. Link Power Management State Diagram



While in L0 state, the X710-TM4 transitions the transmit lane(s) into the L0s state once the idle conditions are met for a period of time defined as follows.

L0s configuration fields are:

- L0s enable — The default value of the *Active State Link PM Control* field in the PCIe *Link Control* register is set to 00b (both L0s and L1 disabled). System software can later write a different value into the *Link Control* register.
- L0s exit latency (as published in the *L0s Exit Latency* field of the *Link Capabilities* register) is loaded from/to the GLPCI_PMSUP.L0S_EXIT_LAT NVM/register field. Separate values are loaded when the X710-TM4 shares the same reference PCIe clock with its partner across the link, and when the X710-TM4 uses a different reference clock than its partner across the link. The X710-TM4 reports whether it uses the slot clock configuration through the *PCIe Slot Clock Configuration* bit loaded from/to the GLPCI_PMSUP.SLOT_CLK NVM/register bit.
- L0s acceptable latency (as published in the *Endpoint L0s Acceptable Latency* field of the *Device Capabilities* register) is loaded from the GLPCI_PMSUP.L0S_ACC_LAT NVM/register field.

The X710-TM4 transitions the link into L0s state once the PCIe link has been idle for a period of time defined in the *l0s_idle_timer* field of PMIDLTMR register that is loaded from RO PCIe LCB module in the NVM. The X710-TM4 transitions the link into the L1 state once the PCIe link has been in the L0s state for a further period as defined in the *l1_idle_timer* register field of the same register.

The following NVM fields control L1 behavior:

- L1 enable — Indicates support for ASPM L1 in the PCIe configuration space (loaded into the *Active State Link PM Support* field).
- L1 exit latency — Defines the L1 active exit latency. The default value is loaded from/to the GLPCI_PMSUP.L1_EXIT_LAT NVM/register field.
- L1 acceptable latency — Defines the L1 active acceptable exit latency. The default value is loaded from/to the GLPCI_PMSUP.L1_ACC_LAT NVM/register field.

5.2.3 Power States

The X710-TM4 supports the D0 and D3 architectural power states as described earlier. Internally, the X710-TM4 supports the following power states:

- D0u (D0 un-initialized) — An architectural sub-state of D0.
- D0a (D0 active) — An architectural sub-state of D0.
- D3 — Architecture state D3hot.
- Dr — Internal state that contains the architecture D3cold state. Dr state is entered when *PE_RST_N* is asserted, a PCIe in-band reset is received or if the function is disabled.

Figure 5-2 shows the power states and transitions between them.

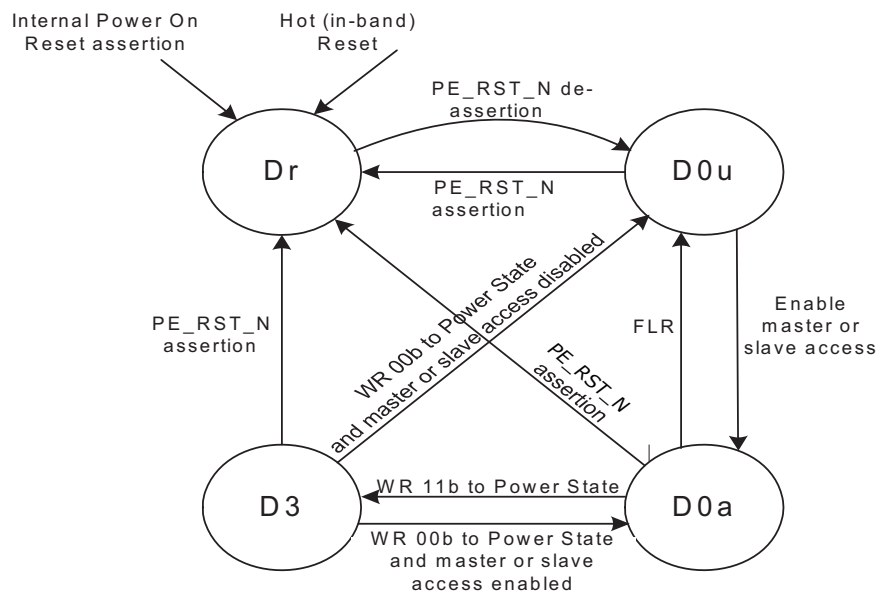


Figure 5-2. Power Management State Diagram

5.2.3.1 D0_{uninitialized} State

The D0_u state is an architectural low-power state. In this state, the device is out of reset (conventional and FLR) but did not complete the enumeration and configuration stage.

5.2.3.1.1 Entry to a D0_u State

D0_u is reached from either the Dr state (after de-assertion of internal *PE_RST_N* and auto load of hardware configuration completes) or the D3_{hot} state (by configuration software writing a value of 00b to the *Power State* field of the PCI PM registers while master and slave accesses are disabled). De-assertion of internal *PE_RST_N* causes the entire state of the X710-TM4 to be cleared except for bits defined as sticky in configuration space. PCIe configuration is loaded from the NVM, followed by establishment of the PCIe link and enumeration. Once this is done, configuration software can access the X710-TM4.

5.2.3.2 D0_{active} State

A The X710-TM4 function enters the D0_{active} state each time any single or combination of the function’s *Memory Space Enable*, *I/O Space Enable*, or *Bus Master Enable* bits have been enabled by system software in the PCI Command configuration register.

In this state it can transmit and receive packets if properly configured by the software device driver. The PHY is enabled or re-enabled by the software device driver to operate/auto-negotiate to full line speed/power if not already operating at full capability.

**Note:**

1. Wake behavior:
 - a. An APM wake event (PM_PME message) due to receiving a Magic packet is not generated when the function is in the D0 active state.
 - b. Any APM wake up previously active remains active when moving from D3 to D0. For example, APM enable is loaded from the NVM and is not reset by any reset but POR.
 - c. WAKE# pin is never toggled for an APM wake event when a function is in D0.
2. If APM wake is required in D3, the software driver should not disable APM wake-up via clearing the PFPM_APM.APME bit on entry into D0. Otherwise, APM wake following a system crash and entry into S3, S4 or S5 system power management state is not enabled. Following entry into D0, the software device driver can activate the wake-up filters by writing to the Wake Up Filter Control (PFPM_WUFC) register.

5.2.3.2.1 Entry to D0a State

D0a is entered from either the D0u state (by writing a 1b to the *Memory Access Enable* or the *I/O Access Enable* bit or the *BME* bit of the PCI Command configuration register) or from the D3_{hot} state (by configuration software writing a value of 00b to the *Power State* field of the PCI PM configuration registers while master or slave accesses is enabled). The receive and transmit flows of the appropriate LAN function are enabled.

5.2.3.3 D3 State (PCI-PM D3hot)

The X710-TM4 functions can transition to D3 when the system writes a 11b to the *Power State* field of the Power Management Control/Status Register (PMCSR). Any wake-up filter settings that were enabled before entering this state are maintained. The X710-TM4 does not clear any bit in the PCIe configuration space of a function during the function's transition to the D3 state. While in D3, the appropriate function of the X710-TM4 does not generate master cycles.

Configuration and message requests are the only TLPs accepted by a function in the D3hot state. All other received requests must be handled as unsupported requests, and all received completions are handled as unexpected completions. If an error caused by a received TLP (such as an unsupported request) is detected while in D3hot, and reporting is enabled, the link must be returned to L0 if it is not already in L0 and an error message must be sent. See section 5.3.1.4.1 in the PCIe Base Specification.

5.2.3.3.1 Entry to D3 State

Transition to the D3 state is through a configuration write to the *Power State* field of the PMCSR PCIe configuration register.

Prior to transition from D0 to the D3 state, the software device driver disables scheduling of further tasks to the X710-TM4; it masks all interrupts, it does not write to the Transmit Descriptor Tail register or to the Receive Descriptor Tail register.

If wake up capability is needed, system software should enable wake capability by setting the *PME_En* bit in the *PMCSR* PCIe configuration register of the PF to 1b. After wake capability has been enabled, the software device driver should set up the required wake up functionality using the flow described in [Section 5.4.5](#) prior to the D3 transition.



Note:

1. The PMCSR.PME_En bit setting can be overridden via the PFPM_APM.APME bit.
2. If operation during D3_{cold} is required, even when wake capability is not required (such as for manageability operation), the system should also set the *Auxiliary (AUX) Power PM Enable* bit in the PCIe Device Control register.

If all PCI functions are programmed into the D3 state, the X710-TM4 attempts to bring its PCIe link into the L1 link state. As part of the transition into the L1 state, the X710-TM4 suspends scheduling of new host TLPs and MCTP over PCIe VDMs are not suspended. the X710-TM4 waits for the completion of all previous TLPs it has sent. Any receive packets that have not been transferred into system memory are kept in the device (and discarded later on D3 exit). Any transmit packets that have not been sent can still be transmitted (assuming the Ethernet link is up).

In preparation for a possible transition to D3_{cold} state, the software device driver might disable up to three LAN ports out of four (LAN disable) and/or transition the remaining link(s) to GbE or 100 Mb/s speed (if supported by the network interface).

5.2.3.3.2 Exit from D3 State

A D3 state is followed by either a D0u state (in preparation for a D0a state) or by a transition to Dr state (PCI-PM D3_{cold} state). To transition back to D0u, the system writes a 00b to the *Power State* field of the PMCSR. Transition to Dr state is through *PE_RST_N* assertion.

The *No_Soft_Reset* bit in the PMCSR in the X710-TM4 is always set to 1b, to indicate that the X710-TM4 does not perform an internal reset on transition from D3_{hot} to D0 so that transition does not disrupt the proper operation of other active functions. Software is not required to re-initialize the function's configuration space after a transition from D3_{hot} to D0 (the function stays in the D0 state).

The function is reset if the link state transitioned to the L2/L3 ready state, on transition from D3_{cold} to D0, if FLR is asserted or if transition D3_{hot} to D0 is caused by asserting PCIe reset (*PE_RST_N* pin).

5.2.3.4 Dr State (D3_{cold})

Transition to the Dr state is initiated on several occasions:

- On system power up — Dr state begins with the assertion of the internal power detection circuit (*LAN_PWR_GOOD*) and ends after the de-assertion of *PE_RST_N* and completion of the NVM auto load.
- On transition from any state — During operation, the system can assert *PE_RST_N* at any time. In an ACPI system, a system transition to the G2/S5 state causes a transition to Dr state.

Any wake up settings that were enabled before entering this reset state are maintained.

The system can maintain *PE_RST_N* asserted for an arbitrary time. The de-assertion (rising edge) of *PE_RST_N* causes a transition to D0u state.

While in the Dr state, the X710-TM4 can maintain functionality (for WoL or manageability) or can enter a Dr Disable state (if no WoL and no manageability) for minimal device power. The Dr disable mode is described in the following section.



5.2.3.4.1 Dr Disable Mode

The X710-TM4 enters a Dr disable mode on transition to D3cold state when it does not need to maintain any functionality. The conditions to enter either state are:

- The device (all PCI functions) is in Dr state.
- APM WOL is inactive for all PCI functions.
- Pass-through manageability is disabled.

Entry into Dr disable is usually done on assertion of PCIe *PE_RST_N*. It can also be possible to enter Dr disable mode by reading the NVM while already in Dr state. The usage model for this later case is on system power up, assuming that manageability and wake up are not required. Once the device enters Dr state on power up, the NVM is read. If the NVM contents determine that the conditions to enter Dr disable are met, the device then enters this mode (assuming that PCIe *PE_RST_N* is still asserted).

Exit from Dr disable is through de-assertion of PCIe *PE_RST_N*.

If Dr disable mode is entered from D3 state, the platform can remove the X710-TM4 power. If the platform removes the X710-TM4 power, it must remove all power rails from the device if it needs to use this capability. Exit from this state is through power-up cycle to the X710-TM4.

Note: The state of the TX_DIS (Optical module TX disable connected to SDP pins) or any of the other SDP pins is undefined once power is removed from the device.

5.2.3.4.2 Entry to Dr State

Dr entry on platform power up is as follows:

- Asserting the internal power detection circuit (LAN_PWR_GOOD). Device power is kept to a minimum by keeping the network interfaces in low power.
- The NVM is then read and determines device configuration.
- If the *APM Enable* bit in the NVM is set, then APM wake up is enabled (for each port independently).
- If the *MNG Enable* bit in the NVM word is set, pass-through manageability is enabled.
- Each of the LAN ports can be enabled if required for WoL or manageability. See [Section 5.3](#) for exact condition to enable a port.
- The PCIe link is not enabled in the Dr state following system power up (since *PE_RST_N* is asserted).

Entry to the Dr state from the D0a state is by asserting the *PE_RST_N* signal. An ACPI transition to the G2/S5 state is reflected in a device transition from the D0a to Dr state. The transition can be orderly (such as a user selected a shut down operating system option), in which case the software device driver can intervene. Or, it can be an emergency transition (like power button override), in which case, the software device driver is not notified.

Transition from the D3 state to the Dr state is done by asserting *PE_RST_N*. Prior to that, the system initiates a transition of the PCIe link from L1 state to either the L2 or L3 state (assuming all functions were already in D3 state). The link enters the L2 state if *PCI-PM PME* is enabled.



5.3 Network Interfaces Power Management

The X710-TM4 disables a network interface and transitions the interface into low-power state in the following cases:

- All LAN ports are in LAN disable mode as a result of the DEV_DIS_N pin.
- Low power state functionality as described in [Section 5.2.1](#).

When the X710-TM4 is in low-power state, the X710-TM4 asserts the respective TX_DIS pin (Connected to an SDP pin) to enable powering down an external PHY or optical module as well.

5.3.1 Energy Efficient Ethernet (EEE)

The X710-TM4 enters EEE Low Power Idle (LPI) mode on transmit or receive independently whenever the device detects no data is scheduled for transmission, or the link partner indicates no data is pending for reception

EEE LPI mode defined in IEEE Std. 802.3az enables power saving by switching off part of the X710-TM4 functionality when no data needs to be transmitted or/and received. A decision on whether the X710-TM4 transmit path should enter LPI mode or exit LPI mode is done according to the need to transmit. Information on whether a link partner has entered LPI mode is detected by the X710-TM4 and used for power saving in the receive circuitry.

When no data needs to be transmitted a request to enter transmit LPI is issued on the internal xxMII Tx interface causing the backplane PHY to transmit sleep symbols for a predefined period of time followed by a quiet period. During LPI, the backplane PHY periodically transmits refresh symbols that are used by the link partner to update adaptive filters and timing circuits in order to maintain link integrity. This quiet-refresh cycle continues until transmission of normal inter-frame encoding on the internal xxMII Tx interface. The backplane PHY communicates to the link partner the move to link active state by sending wake symbols for a predefined period of time. The PHY then enters a normal operating state where data or idle symbols are transmitted.

In the receive direction, entering LPI mode is triggered by receiving sleep symbols from the link partner. This signals that the link partner is about to enter LPI mode. After sending the sleep symbols, the link partner ceases transmission. When the link partner initiates the move to LPI, the backplane PHY indicates assert low power idle on the internal xxMII Rx interface and the X710-TM4 receiver disables functionality to reduce power consumption.

[Figure 5-3](#) and [Table 5-1](#) show the general principles of EEE LPI operation on the Ethernet Link.

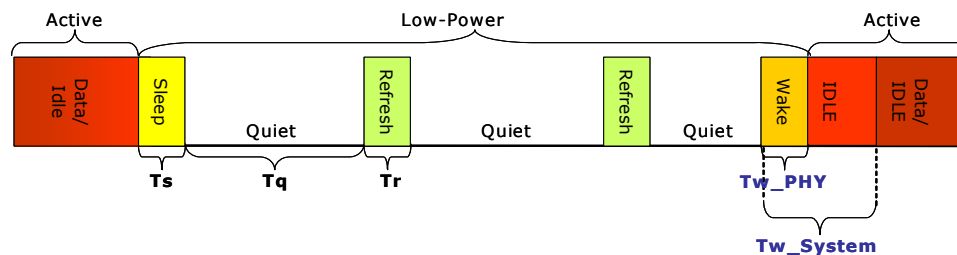


Figure 5-3. Energy Efficient Ethernet Operation

Table 5-1. Energy Efficient Ethernet Parameters

Parameter	Description
Sleep Time (Ts)	Duration PHY sends sleep symbols before going quiet.
Quiet Duration (Tq)	Duration PHY remains quiet before it must wake for refresh period.
Refresh Duration (Tr)	Duration PHY sends refresh symbols for timing recovery and coefficient synchronization.
PHY Wake Time (Tw_PHY)	Minimum duration PHY takes to resume to active state after decision to wake.
Receive System Wake Time (Tw_System_rx)	Wait period where no data is expected to be received to give the local receiving system time to wake up. The X710-TM4 has no plans to use this value for active power reduction.
Transmit System Wake Time (Tw_System_tx)	Wait period where no data is transmitted to give the remote receiving system time to wake up.

5.3.1.1 Conditions to Enter EEE Tx LPI

In the transmit direction entry into EEE LPI mode of operation is triggered when one of the following conditions exist:

- No host transmission is pending, management does not need to transmit, internal transmit buffer is empty, the `PRTPM_EEER.TX_LPI_EN` bit is set to 1b and EEE auto-negotiation completed successfully.
 - When moving out of Tx LPI to transmit a 802.3x flow control or priority flow control frame, the X710-TM4 refrains from re-entering the LPI state until the flow control/priority flow control event indication for any UP in that port was released.
- When `PRTPM_EEEDBG.Force_TLPI` is set, it is software's responsibility to ensure `PRTPM_EEER.TX_LPI_EN` is set and there are no pending transmissions.

When one of the previous conditions to enter Tx LPI state are detected, assert low power idle is transmitted on the internal xxMII interface and the X710-TM4 backplane PHY transmits sleep symbols on the network interface to communicate to the link partner entry into Tx LPI link state. After sleep symbols transmission, the backplane PHY immediately enters low power quiet mode. In this state, the backplane PHY periodically transitions between quiet link state, where link is idle, to send refresh symbols until a request to transition link back to normal (active) mode is transmitted on the internal xxMII Tx interface (See [Figure 5-3](#)).



Note:

PRTPM_EEER.TX_LPI_EN configuration is set by firmware after it finished the EEE timer configurations.

Backplane Ethernet LPI enables each link direction to enter sleep, refresh or wake states asymmetric from the other direction.

EEE LPI status of a X710-TM4 port can be found in the PRTPM_EEE_STAT register.

5.3.1.2 Transition from Tx LPI to Active Link State

The X710-TM4 exits Tx LPI link state and transitions link into active link state when none of the conditions defined in [Section 5.3.1.1](#) exist. To transition into active link state, the X710-TM4 transmits:

1. Normal inter-frame encoding on the internal xxMII Tx interface for a pre-defined link rate dependent period time of `Tw_sys_tx-min` as defined by `PRTPM_EEEC.TW_PHY`. As a result, the backplane PHY transmits wake symbols for a `Tw_phy` duration followed by idle symbols.
2. If the `Tw_System_tx` duration defined in the `PRTPM_EEER.Tw_system` field is longer than `Tw_sys_tx-min`, the X710-TM4 continues transmitting the inter-frame encoding on the internal xxMII interface until the time defined in the `PRTPM_EEER.Tw_system` field has expired, before transmitting the actual data. During this period the backplane PHY continues transmitting idle symbols.

Note: When moving out of Tx LPI to transmit a 802.3x flow control frame, the X710-TM4 only waits the `Tw_phy` duration before transmitting the flow control frame. It should be noted that even in this scenario actual data is transmitted only after the `Tw_System_tx` time defined in the `PRTPM_EEER.Tw_system` field has expired.

Note: When moving out of Tx LPI to transmit a 802.3x flow control or priority flow control frame, the X710-TM4 refrains from re-entering LPI state until the flow control/priority flow control event indication for any UP in that port was released.

5.3.1.3 EEE Auto-negotiation

Backplane auto-negotiation provides the capability to negotiate EEE capabilities with the link partner using the next page mechanism defined in IEEE Std. 802.3-2018 Annex 73A. IEEE Std. 802.3-2018 Auto-negotiation is performed at power up, on command from software, upon detection of a PHY error or following link re-connection.

During the link establishment process, both link partners indicate their EEE capabilities using IEEE Std. 802.3-2018 auto-negotiation. If EEE is supported by both link partners for the negotiated PHY type then the EEE function might be used independently in either direction.

When operating in 1000BASE-KX or 10GBASE-KR link, EEE capabilities advertised during auto-negotiation can be modified via the backplane PHY registers.

Auto-negotiation of EEE capabilities is also supported when the X710-TM4 ports are configured to operate in SGMII mode (see [Section 5.3.1.7.2](#)).



5.3.1.4 EEE Link Level (LLDP) Capabilities Discovery

When operating in 1000BASE-KX or 10GBASE-KR link and auto-negotiation is resolved with EEE support, the X710-TM4 supports LLDP negotiation via software or firmware, using the EEE IEEE802.1AB Link Layer Discovery Protocol (LLDP) *Type, Length, Value* (TLV) fields defined in IEEE Std. 802.3-2018 clause 78 and clause 79. LLDP negotiation enables negotiation of increased system wake time (Transmit T_w and Receive T_w) to enable improving system energy efficiency.

During pre-boot, LLDP is done by firmware according to the following flow:

1. EMP determines whether EEE has to be enabled or disabled over a LAN port by reading the *EEE Capability* bit in the corresponding PHY capability data structure in the EMP Settings NVM module.
2. If enabled, EMP loads the settings for the EEE LLDP variables from the EMP settings module. The relevant variables are: $T_{w_sys_tx_local}$ (Transmit T_w), $T_{w_sys_rx_local}$ (Receive T_w), and $T_{w_sys_rx_fb}$ (Fallback T_w). These fields can be modified post boot using admin commands. For more information on LLDP protocol, activation and flow refer to [Section 7-12](#).
3. If enabled, the PHY goes to an auto-negotiation cycle to define the EEE support with the link partner.
4. Once done, EEE is operational on both Rx and Tx (if enabled on Tx) using default values.

It is assumed that the default latency values are sufficient for system support.

5.3.1.4.1 LLDP Negotiation Actions

Following negotiation of a new system wake time via EEE LLDP negotiation, the `PRTPM_EEER.Tw_system` field should be updated by firmware (when firmware does the LLDP) or by software by sending a Set PHY Config command (see [Section 3.2.12.1.1](#)) via the admin queue (when software does the LLDP) with the negotiated transmit T_w time value, to increase the duration where idle symbols are transmitted following move out of the EEE Tx LPI state before actual data can be transmitted.

Note: When the firmware LLDP agent is turned off, the EEE agent is disabled as well.

5.3.1.5 Programming for Backplane EEE Operation

The following fields should be programmed to enable EEE operation on a LAN port when operating in 1000BASE-KX or 10GBASE-KR modes:

1. Modify PHY registers if default EEE advertised auto-negotiation values need to be modified.
2. Set the `PRTPM_EEER.TX_LPI_EN` bit to 1b to enable EEE LPI support, if the result of auto-negotiation at the specified link speed enables entry to LPI by sending a Set PHY Config command (See [Section 3.2.12.1.1](#)) via the admin queue.
 - a. `PRTPM_EEER.TX_LPI_EN` is set by firmware after `PRTPM_EEETXC.TW_PHY` is properly assigned by firmware.
3. On every link up indication, firmware should set the `PRTPM_EEEC.TW_PHY` bit with the proper value from the IEEE standard augmented by the per technology value defined in `GLPM_EEE_SU/GLPM_EEE_SU_EXT`. Once all EEE settings are completed, firmware should assert the `PRTPM_EEEFWD.EEE_FW_CONFIG_DONE` bit indicating to hardware that the configuration cycle completed.



Note: In contrast to previous products, PRTEEEER.TX_LPI_EN is not used to gate the activation of EEE in the Tx side and the indication must be used by the EMP firmware to gate the setting of PRTPM_EEEFWD.EEE_FW_CONFIG_DONE. This means that before setting PRTPM_EEEFWD.EEE_FW_CONFIG_DONE, EMP firmware must check the value of PRTEEEER.TX_LPI_EN and gate the setting of PRTPM_EEEFWD.EEE_FW_CONFIG_DONE if it is clear. Also, when a Set PHY Config command is used to clear/set PRTEEEER.TX_LPI_EN, EMP firmware must clear/un-gate the setting of PRTPM_EEEFWD.EEE_FW_CONFIG_DONE.

Following EEE activation, an EEE LLDP negotiation cycle should be initiated and the appropriate registers should be programmed (See Section 5.3.1.4).

Note:

1. The X710-TM4 waits for at least one second (depending on value in PRTPM_EEEC.TX_LU_LPI_DLY field) following auto-negotiation (due to reset, link disconnect or link speed change) and link-up indication before enabling link entry into the EEE Tx LPI state to comply with the IEEE Std. 802.3-2018 specification.
2. Fast retrain and local/remote fault indication are not considered link disconnect and do not require waiting for one second before enabling link to enter into EEE Tx LPI.

5.3.1.6 EEE Statistics

The X710-TM4 supports reporting a number of EEE LPI Tx and Rx events via the PRTPM_RLPIC and PRTPM_TLPIC registers.

When operating at 2.5 Gb/s or 5 Gb/s speeds, native EEE mode uses the same mechanism as described in the link flow control section. Meaning, the KR Rx link between the X710-TM4 MAC and the BASE-T PHY remains in an active state.

The EEE indication and LPI counters should be read from the BASE-T PHY and not from the X710-TM4 MAC. Read register 0x3.0001 (PCS Status 1) and get the indication according to the following table.

Idle Indication	Description
Bit 9: Tx LP idle indication.	1b = Tx is currently receiving LP idle. 0b = LP idle not received.
Bit 8: Rx LP idle indication.	1b = Rx is currently receiving LP idle. 0b = LP idle not received.

In order to get the LPI counters, use the command handler GET_EEE_STATISTIC via the user defined AQ command (opcode 0x626).

Note:

- The TW_SYSTEM field should initialize to a default value immediately after link down (see Section 10.2.1.4.2).
- The minimum wake time for the speeds without any snaking should be taken from table 78-4 in the IEEE Std. 802.3-2018 standard.
 - 100TX - 30 μs
 - 1G - 16.5 μs
 - 2.5G - 4*7.36 μs



- 5G - 2*7.36 μ s
- 10G - 7.36 μ s

Connecting two EEE links back-to-back on the X710-TM4 MAC side, requires an increase in wake time. When operating at 1 GbE EEE, the wake time should be increased by a factor of two.

If EEE is disabled, restore all default values to the registers previously mentioned.

5.3.1.7 External PHY EEE Support

The X710-TM4 supports EEE when connected to an external PHY.

5.3.1.7.1 Programming for External PHY EEE Operation

The following fields should be programmed to enable EEE operation on a LAN port when connected to an external copper PHY:

1. After verifying that the external copper PHY supports EEE. Relevant external PHY EEE related registers can be modified (access varies based on the PHY), if the default EEE advertised auto-negotiation values need to be updated.
2. Set the PRTPM_EEER.TX_LPI_EN bit to 1b to enable EEE LPI support, if the result of auto-negotiation at the specified link speed enables entry to LPI by sending a Set PHY Config Command (See [Section 3.2.12.1.1](#)) via the admin queue.
 - a. PRTPM_EEER.TX_LPI_EN is set by firmware after PRTPM_EEETXC.TW_PHY is properly assigned by firmware.
3. On every link up indication (internal and external) firmware should set the PRTPM_EEEC.TW_PHY bit with the proper value from the IEEE standard augmented by the per technology value defined in GLPM_EEE_SU/GLPM_EEE_SU_EXT.
4. When operating in XFI or SFI operating modes, the corresponding PRTMAC_PCS_AN_CONTROL4 register bits should be set to 1b to enable EEE operation on the interface regardless of the (external) PHY auto-negotiation results. This is required for external PHYs where EEE auto-negotiation results are not reported to the MAC over the MAC-PHY interface.

Following EEE activation, an EEE LLDP negotiation cycle should be initiated and the appropriate registers should be programmed (See [Section 5.3.1.4](#)).

Note:

1. The X710-TM4 waits for at least one second (depending on value in PRTPM_EEEC.TX_LU_LPI_DLY field) following auto-negotiation (due to reset, link disconnect or link speed change) and link-up indication before enabling link entry into the EEE Tx LPI state to comply with the IEEE Std. 802.3-2018 specification.
2. Fast retrain and local/remote fault indication are not considered link disconnect and do not require waiting for one second before enabling link to enter into EEE Tx LPI.

5.3.1.7.2 SGMII EEE Operation

When port is configured for SGMII operating mode the X710-TM4 detects external PHY EEE support during SGMII Auto-negotiation when bit 9 of the received Auto-negotiation Link Control word is set to 1b (See [Section 3.2.7.4.2](#) for additional information).



If the external PHY reports support of IEEE Std. 802.3az operation, the `PRTPM_EEER.TX_LPI_EN` bit is set to 1b and External PHY operating mode is either Full duplex 100BASE-TX or Full-duplex 1000BASE-T, the X710-TM4 will transmit IEEE Std. 802.3az clause 36 /LI/ PCS ordered sets to initiate move of External PHY into EEE TX LPI state when conditions to enter EEE TX LPI state exist (See [Section 5.3.1.1](#)). When entering the EEE LPI state the X710-TM4 powers down relevant circuitry to reduce power consumption.

If the external PHY reports support of IEEE Std. 802.3az operation, the External PHY operating mode is either Full duplex 100BASE-TX or Full-duplex 1000BASE-T, the X710-TM4 will detect reception of IEEE Std. 802.3az clause 36 /LI/ PCS ordered sets and move internal circuitry into EEE RX LPI state.

Note: Even if external PHY does not report support of EEE during Auto-negotiation the X710-TM4 can be forced to enable EEE operation by setting the corresponding bits in `PRTMAC_PCS_AN_CONTROL4` register.

5.3.1.7.3 SFI or XFI EEE Operation

When a port is configured for SFI or XFI operating mode and the external 10GBASE-T PHY supports IEEE Std. 802.3az operation, the corresponding bits in `PRTMAC_PCS_AN_CONTROL4` register should be set to 1b to enable EEE operation on the SFI or XFI interface (For further information on SFI or XFI operation see [Section 3.2.7.3.2](#)).

If the external 10GBASE-T PHY connected to the SFI or XFI interface supports IEEE Std. 802.3az EEE operation, the corresponding bits in `PRTMAC_PCS_AN_CONTROL4` register are set to 1b and the `PRTPM_EEER.TX_LPI_EN` bit is set to 1b, the X710-TM4 transmits IEEE Std. 802.3az clause 49 /LI/ PCS control characters to initiate a move of the external PHY into the EEE Tx LPI state when conditions to enter EEE Tx LPI state exist (See [Section 5.3.1.1](#)). When entering the EEE LPI state, the X710-TM4 powers down relevant circuitry to reduce power consumption.

1. If the external 10GBASE-T PHY connected to the SFI or XFI interface supports IEEE Std. 802.3az EEE operation, the corresponding bits in `PRTMAC_PCS_AN_CONTROL4` register are set to 1b, the X710-TM4 detects receiving IEEE Std. 802.3az clause 49 /LI/ PCS control characters and moves internal circuitry into the EEE Rx LPI state.

5.3.2 Dx Low Power Link Up (LPLU)

Auto-negotiation enables establishing link speed at the Highest Common Denominator (HCD). When all PCIe functions belonging to a port are in the Dx low power state and the X710-TM4 is connected to an auxiliary power supply and main power is down, power saving might be more important than performance.

The X710-TM4 supports a mode where it auto-negotiates to the Lowest Common Denominator (LCD) link speed (such as the lowest commonly supported speed) in low power states.

Initially in this mode, the X710-TM4 attempts to negotiate to the lowest link rate. If the link partner doesn't support the lowest link rate, the X710-TM4 auto-negotiates to the lowest link rate advertised by the link partner.

5.3.2.1 LPLU State Entry/Exit Logic

Operation is configured per port using the `PRTPM_GC.LCDMP`, `PRTPM_GC.RATD` and `PRTPM_GC.MNG_VETO` register bits.



The software device driver can enable Dx low power backplane auto-negotiation when operating in a single PF per port mode using the Set PHY Config Command (See Section 3.2.12.1.1) via the admin queue. The LPLU admin state is reflected by EMP to all the PFs attached to the port into PRTPM_GC.LPLU_ASSERTED bit.

The X710-TM4 can initiate auto-negotiation without direct driver command in the following cases:

- When the state of MAIN_PWR_OK pin changes.
- When the PRTPM_GC.MNG_VETO bit value changes.
- When all functions of a port transition from the D0 (D0a or D0u) state to the Dx state, or from the Dx state to the D0 state.

Note: The MAIN_PWR_OK pin is de-asserted at least 10 ms before power drops below its 90% value. This allows enough time to the PHY to drop the link and restart auto-negotiation before auxiliary power takes over. Note that since auto-negotiation is restarted, the PHY power consumption might already cut down.

Figure 5-4 and Figure 5-5 show the X710-TM4 behavior when entering a low power state and during power state transitions.

Note: The following flow might cause the VETO bit not to be honored when transitioning from low power state to D0.

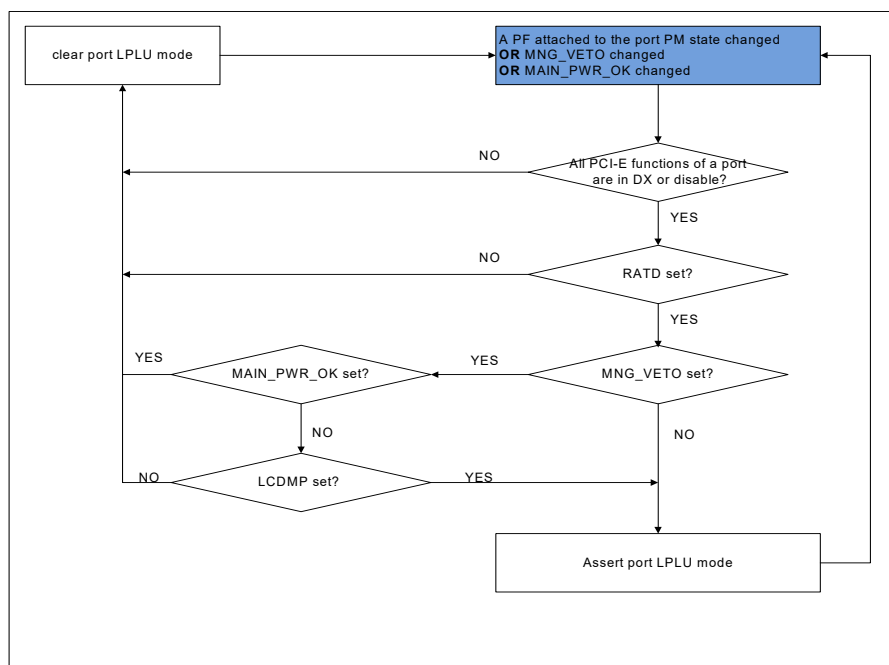


Figure 5-4. LPLU Assert/Clear When Entering Dx Power Down Mode

5.3.2.2 LPLU Related Link Speed Change

Figure 5-5 shows the flow for changing the link speed when entering/existing the LPLU state.

The X710-TM4 attempts to lower the link to the lowest commonly supported speed when entering the LPLU state. Entering the LPLU state and switching to a lower link speed is done by clearing the relevant bits in the auto-negotiation advertised capabilities in the (internal/external) PHY configuration (advertising only the lowest common speed), and triggering an auto-negotiation process. The device is expected to link at a lower speed as higher speeds are not advertised as supported.

Similarly, exiting the LPLU state and switching to a higher link speed is done by setting the relevant bits in the auto-negotiation advertised capabilities in the PHY configuration, and restarting auto-negotiation. The device is now expected to link at the highest common speed.

Note: Given the previous, a temporary link loss (and a disruption to traffic) is expected when entering/exiting the LPLU state.

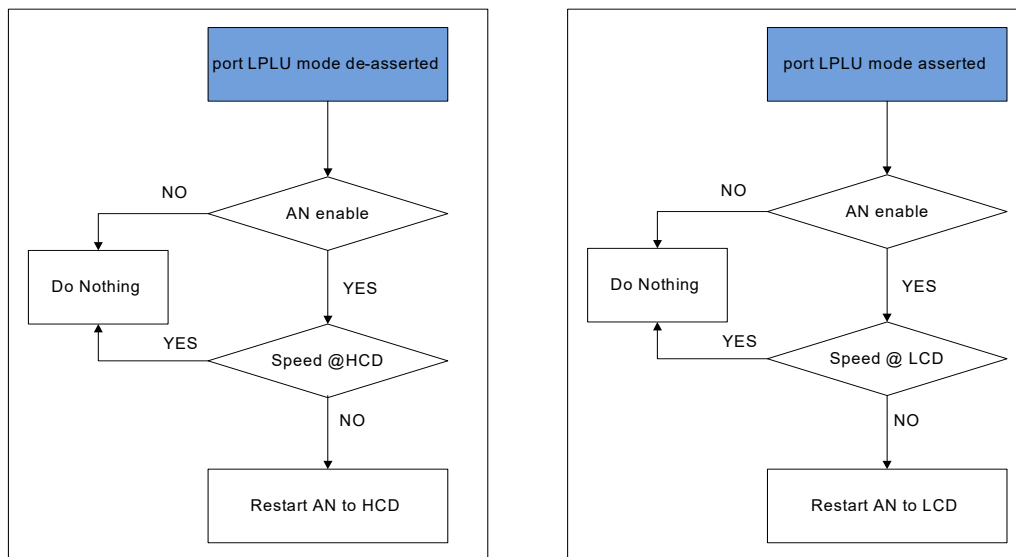


Figure 5-5. Link Speed Change on LPLU Assert/Clear

5.4 Wake Up

The X710-TM4 supports wake up functionality in accordance to the APM and ACPI specifications. The following list provides a high-level view on the functionality supported in each specification while the following section provide more details and elaboration.

1. Advanced Power Management (APM) wake up (enabled via PFPM_APM.APME), which means support for wake from S5 (soft off) states using the well-known Magic Packet wake packet format.
2. ACPI wake up capabilities (enabled via PMCSR.PME_EN):
 - a. Link State Change (LNKC) wake up, which means the ability to wake up the host from S5 when a LAN link state goes up.
 - b. Magic packet (MAG) wake up, which means support for wake from S5 (soft off) states using the well-known Magic Packet wake packet format.
 - c. Firmware reset (FW_RST_WK) wake up, which means the ability to wake up the host from S5 when an EMPR event occurs.



- d. Manageability (MNG) wake up, which means the ability for the EMP to wake up the host from S5, on its own initiative. No usage model actually defined for this option.

These wake up mechanisms are basically controlled by the PF via the corresponding bit in PFPM_WUFC and PFPM_WUS registers. It suffices that one PF enables a wake-up mechanism for the wake up to be enabled for the concerned port.

5.4.1 Advanced Power Management Wake Up

This feature has existed in the 10/100 Mb/s NICs for several generations. Systems were activated from S3, S4 or S5 low power states after receiving a Magic packet that is a broadcast or unicast packet with an explicit data pattern. After receiving a Magic packet, the X710-TM4 can wake up the system by also asserting the PCIe PE_WAKE_N signal and optionally via an in-band PM_PME message.

An APM wake-up operation is controlled by the per PF APM Control (PFPM_APM) register. At power up, the X710-TM4 reads the *APM Enable* bit from the NVM into the *APM Enable* (PFPM_APM.APME) bit, this bit is used to enable WoL by overriding the *PME_En* bit. This bit controls enabling of the APM wake up. Software enables or disables APM wake up using the flows described in [Section 5.4.5.1](#) and [Section 5.4.5.2](#), respectively.

When APM wake up is enabled, the X710-TM4 checks all incoming packets passing L2 filters (BCST, MCST, UCST) for Magic packets. See [Section 5.4.3.1.1](#) for a definition of Magic packets. BCST and UCST (according to PFPM_SAL/H that are loaded from the NVM and properly assigned by firmware to the PRTPM_SAL/H registers) Magic packet filtering is enabled by default. To enable promiscuous MCST Magic packets, WoL PRTPM_SAH.MC_MAG_EN needs to be set to 1b.

Once the X710-TM4 receives a matching Magic packet, and the PFPM_APM.APME bit is set to 1b it executes the ACPI flow.

Note: When the PFPM_APM.APME bit is set a wake event is issued (PE_WAKE_N pin is asserted and a PM_PME PCIe message is issued) even if the PMCSR.PME_En bit in configuration space is cleared. To enable disabling of system wake up when PMCSR.PME_En is cleared, the software device driver should clear the PFPM_APM.APME bit after power-up or PCIe reset.

5.4.2 ACPI Power Management Wake Up

The X710-TM4 supports ACPI power management-based wake up. It can generate system wake-up events from a number of sources:

- Detection of a change in network link (cable connected or disconnected).
- Firmware reset (FW_RST_WK) wake up, which means the ability to wake up the host from S5 when an EMPR event occurs.
- Magic packet (MAG) wake up, which means support for wake from S5 (soft off) states using the well known Magic packet wake packet format.
- Manageability (MNG) wake up, which means the ability for the EMP to wake up the host from S5, on its own initiative. No usage model actually defined for this option.

Note: ACPI wake-up operation is controlled by the per PF Wake Up Filter Control (PFPM_WUFC) register and wake-up status is reported in the per PF Wake Up Status (PFPM_WUS) register. In the PCIe configuration space, the PMCSR register supports wake up. Software enables ACPI wake up using the flow described in [Section 5.4.5](#).



5.4.3 Wake-Up Filters

The X710-TM4 supports issuing a wake-up indication to the host after receiving non-errored packets when the device is in the D3 low power state or in Dr with WoL enabled, using Magic packet filters set from NVM.

Support of wake up in Dr is only available if the *AUX_PWR* bit is asserted.

5.4.3.1 Wake Up Filter Types

The following packet types are detected per PF by the X710-TM4 as a possible wake up.

5.4.3.1.1 Magic Packet

Magic packets are defined in:

<http://support.amd.com/TechDocs/20213.pdf> as:

“Once the LAN controller has been put into the Magic Packet mode, it scans all incoming frames addressed to the node for a specific data sequence. This sequence indicates to the controller that this is a Magic Packet frame. A Magic Packet frame must also meet the basic requirements for the LAN technology chosen, such as Source Address, Destination Address (which may be the receiving station's IEEE address or a Multicast address which includes the Broadcast address), and CRC. A magic packet must also be a valid non error L2 packet. The specific data sequence consists of 16 repetitions of the IEEE address of this node, with no breaks or interruptions. This sequence can be located anywhere within the packet, but must be preceded by a synchronization stream. The synchronization stream allows the scanning state machine to be much simpler. The synchronization stream is defined as 6 bytes of 0xFF. The device will also accept a Broadcast frame, as long as the 16 repetitions of the IEEE address match the address of the machine to be awakened.”

The X710-TM4 expects the destination address to either:

- Be the broadcast address (FF.FF.FF.FF.FF.FF)
- Match the value of the port L2 Ethernet destination MAC address (DA) set by firmware to the relevant filters PRTPM_SAL/H.

On power up firmware loads the PRTPM_SAL/H addresses from the NVM. Each address entry contains:

- MAC address for Magic packet filtering
- Address Valid (AV) indication
- Promiscuous multicast magic filtering enable (MC) indicating the L2 destination address can be any multicast address
- The PF number (*PF_NUM*) to be reported if a Magic packet matches this filter

Firmware uses the PFGEN_PORTNUM registers to map the per PF MAC address to the relevant ports.



The X710-TM4 searches for the contents of the port's Ethernet destination MAC address receive address as the embedded IEEE address. It considers any non-0xFF byte after a series of at least six 0xFFs to be the start of the IEEE address for comparison purposes. For example, it catches the case of seven 0xFFs followed by the IEEE address). As soon as one of the first 96 bytes after a string of 0xFFs don't match, it continues to search for another set of at least six 0xFFs followed by the 16 copies of the IEEE address later in the packet. Note that this definition precludes the first byte of the destination address from being FF.

A Magic packet's destination address must match the address filtering enabled in the configuration registers with the exception that broadcast packets are considered to match even if broadcast packet reception is not enabled. If APM wake up (wake up by a Magic packet) is enabled in the NVM, the X710-TM4 starts up with the Ethernet MAC destination address loaded from the NVM. This enables the X710-TM4 to accept packets with the matching IEEE address before the software device driver loads.

Table 5-2. Magic Packet Structure

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC header – processed by main address filter.
6	6	Source Address		Skip	
12	T=(0/4/8)	Possible S-TAG		Skip	
12 + T	S=(0/4)	Possible VLAN Tag		Skip	
12 + T + S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + T+ S + D	2	Type		Skip	
Any	6	Synchronizing Stream	FF*6+	Compare	
any+6	96	16 Copies of Node Address	A*16	Compare	Compared to relevant station addresses (PRTPM_SAH, PRTPM_SAL) registers.

5.4.4 Wake Up and Virtualization

When operating in a virtualized environment, all wake-up capabilities are managed by a single entity (such as the VMM or an IOVM). In an IOV architecture, the physical (PF) driver controls wake up and none of the Virtual Machines (VMs) has direct access to the wake-up registers. The wake-up registers are not replicated per VF.

5.4.5 Wake-Up Flows

5.4.5.1 Wake-Up Enable Flow

A WoL register set exists in each PF. Each set consists of the following wake-up filters and registers:

1. *PFPM_WUC* - Wake Up Control Register.
2. *PFPM_WUS* - Wake Up Status Register.



On power up, values loaded from the NVM into the following per PF register define the enabling of APM wake-up functionality:

1. The NVM *APM Enable* bit is loaded to the PFPM_APM.APME register bit to enable:
 - a. Detection of a Magic packet destined to the PF.

Note: The *EN_APM_D0* bit also enables the wake functionality in ACPI mode while the device is in D0. Therefore, if any of the ACPI wake filters are enabled (See [Section 5.4.2](#)) and wake events in D0 are not desired, *EN_APM_D0* needs to be cleared by software. Similarly, if software needs to enable wake events in D0, the *EN_APM_D0* bit needs to be set by software on top of enabling the relevant ACPI wake filters.

5.4.5.2 Wake-Up Disable Flow

Once the X710-TM4 wakes the system following transition to D0, the software device driver might disable WoL directly by:

1. Clearing all the pending wake-up status bits in the Wake Up Status (*PFPM_WUS*) register.
2. Clearing the relevant bits in the *PFPM_WUC* register.

Note: This flow is not used by normal Intel drivers.

5.4.5.3 ACPI Wake-up Flow

Once wake up is enabled, the X710-TM4 monitors the enabled wake functions. If at least one of the enabled wake events occurs, the X710-TM4 will:

- Sets the *PME_Status* bit in the PFPM_WUS and PCI PMCSR register.
- Sets the proper bit, corresponding to the wake reason in PFPM_WUS:
 - LINKC for link status change.
 - MAG for Magic packet reception.
 - MNG for general manageability wake up.
 - *FW_RST_WK* for EMP reset event.
- If the *PME_En* bit in the PMCSR configuration register is set, asserts *PE_WAKE_N* and/or sends a PM_PME PCIe message.
- Sets one or more of the bits in the PFPM_WUS register. The X710-TM4 sets more than one bit if a packet matches more than one filter.

Note: The *PE_WAKE_N* signal remains asserted, PM_PME messages are periodically sent to the host, and the X710-TM4 ignores any subsequent ACPI wake-up packets and link change events for that function until the operating system either writes a 1b to the *PME_Status* bit of the PMCSR register or writes a 0b to the *PME_En* bit or until de-assertion of *PE_RST_N*.



5.4.6 Reset on LAN (RoL)

Low cost systems without a BMC sometimes require remote boot capability. The X710-TM4 supports two methods for generating a system level reset, via assertion of the PCIe WAKE# pin after receiving a pre-defined packet, for systems that do not require any WoL capability, or for systems that require the PCIe WAKE# pin for WoL by issuing a system level reset after receiving a pre-defined packet via a dedicated GPIO pin when the `GLGEN_GPIO_CTL.PIN_FUNC` is set to RoL mode.

To support RoL, the X710-TM4 should be programmed by the software device driver in the following manner:

- `PFPCI_PM.PME_En` bit should be set to 1b to enable wake up functionality.
- Program the flexible host filter table (`PFPM_FHFT_DATA`, `PFPM_FHFT_MASK`, `PFPM_FHFT_LENGTH`), with the RoL packet.
- When using the WAKE# pin for RoL, the flex filter should be enabled and the action in `PFPM_WUFC` should be set to WoL. In addition, the `GLPCI_CNF.WAKE_PIN_EN` bit should be set to enable the use of the WAKE# pin in all power states.
- When using GPIO pin for RoL, the relevant GPIO control register should be configured to RoL mode through `GLGEN_GPIO_CTL.PIN_FUNC`, the flex filter should be enabled.
- On an RoL event, the GPIO value is driven low and is only asserted back to the default (1b) value according to the `GLPM_WUMC.RoL_Mode` as follows:
 - When the RoL event is cleared by clearing the flex filter event in the `PFPM_WUS` register.
 - After 50 ms from assertion.

Note: The RoL defined GPIO default value can be swapped using `GLPM_WUMC.SRST_PIN_VAL`.

When RoL is configured to operate through GPIO it can be enabled in conjunction to WoL.

WAKE# pin is de-asserted when `PE_RST_N` is asserted (such as reset taking place).

Following a wake-up event, either BIOS or option ROM code must re-enable the `PFPM_WUFC.MAG` bits for the next RoL event.



NOTE: *This page intentionally left blank.*



6.0 Non-volatile Memory Map

Caution: In the tables of this chapter, contents of the NVM Default Value column might not reflect the value programmed in the specific NVM image provided to the customer.

6.1 NVM Organization

6.1.1 Hierarchical NVM Map

Table 6-1 lists in a hierarchical order of all the modules and sub-modules that are defined in the X710-TM4’s NVM map. They are referred to separately as NVM sections. All details concerning these sections are found in Section 6.2.

Note that the root section named Init Module (NVM header) is not listed in Table 6-1. Details about this section can be found in Section 6.1.2.

Table 6-1. Hierarchical NVM Map

Address of Pointer in Parent Section	1 st Level Section	2 nd Level Section	3 rd Level Section	Description
0x06	RO PCIR Registers Auto-load			Contains RO parameters that configure the PCIe transaction layer (note that it is currently empty).
0x0A	RO PCIe LCB			Contains RO parameters that configure the PCIe link layer (LCB unit).
0x37	VLAN Configuration Block			Original factory default MAC addresses defined for LAN.
0x28	Permanent SAN MAC Address			MAC addresses used for SAN.
0x16	PBA Block			The PBA block contains the complete PBA number including the dash and the first digit of the 3-digit suffix.
0x30	PXE Setup Options			Setup options for the PXE driver that is defined per PCIe function.
0x31	PXE Configuration Customization Options			Configuration customization options for the PXE driver that is defined per PCIe function.
0x2F	VPD Module			Vital Product Data (VPD) loaded by an OEM. It contains RO and RW information about a NIC/LOM.
0x17	Boot Configuration Block			Contains the required setup used for boot operations.
0x08	PCIR Registers Auto-load			Default setup to registers and internal memories that load on PCIR events.
0x38	POR Registers Auto-load			Default setup to registers that load on POR events.
0x3C	CORER Registers Auto-load			Default setup to registers and internal memories that load on CORER events.



Table 6-1. Hierarchical NVM Map

Address of Pointer in Parent Section	1 st Level Section	2 nd Level Section	3 rd Level Section	Description
0x3B	GLOBR Registers Auto-load			Default setup to registers that load on GLOBR events.
0x4A	Core Mem Config			Read margin settings to the device's core memories module.
0x48	EMP SR Settings			This section contains the modes of operation of the EMP that must be stored in the shadow RAM.
	0x05	SR PF Allocations		This section contains the allocation of resources per PF that must be stored in shadow RAM.
	0x0F	MNG MAC Addresses		This section contains the Pass Through LAN Ethernet MAC addresses, 4 addresses per port.
	0x10	PF MAC Addresses		This section contains the PF Ethernet MAC addresses, one address per PF.
0x07	Auto Generated Pointers			Pointers to type 1/2 words used by EMP and software. The module must be mapped to shadow RAM word address 0x7D80.
0x3E	PCIe ALT Auto-load			Modified factory default settings to registers that load on PCI reset events. The module must be mapped to shadow RAM word address 0x7E00. Any change performed to the module contents in shadow RAM is not dumped into the Flash memory.
0x03	PCIe Analog			Configures the analog section of the PCIe PHY. Module is authenticated.
0x04	PHY Analog			Configure the analog section of the SerDes PHY. Module is authenticated.
0x09	EMP Global			This section contains two sub-sections: 1. Vendor Specific Settings: Settings for external PHY or modules. This part has a proprietary format in order to enable advanced PHY setting. 2. List of Qualified Modules: Parameter list of up to 16 modules. Per-module list holds OUI, revision and version numbers. These settings are loaded into the external devices via the MDIO interface.
0x0E	Manageability Module Header			This section contains parameters related to the manageability functionality such as connection type and others. It also points to sub-sections configuring the filters and the sideband interfaces.
	0x03	Pass Through Control Words Structure 0		This section contains the initial setting of the pass-through filters for a port. This section is used only in SMBus legacy pass-through mode. This section is repeated per port.
	0x04	Pass Through Control Words Structure 1		
	0x05	Pass Through Control Words Structure 2		
	0x06	Pass Through Control Words Structure 3		
	0x08	Flexible TCO Filter Configuration Structure		



Table 6-1. Hierarchical NVM Map

Address of Pointer in Parent Section	1 st Level Section	2 nd Level Section	3 rd Level Section	Description
	0x07	Sideband Configuration Structure		This section describes the setting of the different pass-through interfaces (SMBus, NC-SI and MCTP).
	0x0B	OEM Section		This section is the header of the OEM specific data identifying the OEM for which this data is defined.
0x48	EMP SR Settings			This section contains the modes of operation of the EMP stored in shadow RAM.
	0x05	PF SR Allocations		This section contains the allocation of resources per PF and the PF MAC address.
0x0F	EMP Settings			This section contains the modes of operation of the EMP.
	0x07	PHY Capability Data Structure 0		The PHY capabilities data structure contains all the parameters to control the internal and external PHY operation. For example, interface type and mode, EEE parameters, etc. Data structure is repeated per port.
	0x08	PHY Capability Data Structure 1		
	0x09	PHY Capability Data Structure 2		
	0x0A	PHY Capability Data Structure 3		
	0x0B	External-to-Internal PHY Mapping 0		This section provides the internal PHY mode that is selected for each external PHY/module counter part.
	0x0C	External-to-Internal PHY Mapping 1		The X710-TM4 provides multiple PHY interface for connecting different types of external PHYs and optical or copper modules. End users can plug in different types of modules into the SFP+ connectors and choose from several connectivity options when using external PHYs. For example a 10GBASE-T PHY could use a KR connection to connect to the X710-TM4.
	0x0D	External-to-Internal PHY Mapping 2		
	0x0E	External-to-Internal PHY Mapping 3		Firmware has to read the PHY or module ID and choose the appropriate interface to operate with that external module. Further, when operating with an external 10GBASE-T PHY, auto-negotiation might result in several speed modes. For example, when a 10GBASE-T PHY can auto-negotiate to 1 GbE. In this case, firmware needs to reduce the internal link to SGMII. This section is repeated per port.
	0x0F	LLDP Configuration		Default settings to the embedded LLDP agent.
	0x14	LLDP OEM TLVs		A set of fixed-structure LLDP TLVs for EMP use.
0x46	2 nd Free Provisioning Area			Free area used as the new bank when updating 8 KB long modules that are mapped outside the shadow RAM.
0x05	Expansion/Option ROM (OROM)			It contains pre-boot code and settings read by BIOS. Pre-boot code is authenticated by BIOS during initialization before being executed by an EMP on update.
0x0B	EMP Image			It contains the EMP processor code.
0x40	1 st Free Provisioning Area			Free area used as the new bank when updating 1160 KB long modules.
0x42	Reserved 4 th Free Provisioning Area			Free area to be used as the new bank when updating future 64 KB long modules mapped outside the shadow RAM.



Table 6-1. Hierarchical NVM Map

Address of Pointer in Parent Section	1 st Level Section	2 nd Level Section	3 rd Level Section	Description
0x44	Reserved 3 rd Free Provisioning Area			Free area to be used as the new bank when updating future 16 KB long modules mapped outside the shadow RAM.
0x49	Feature Configuration			Holds the currently used set of selections for the current NVM (Shadow RAM)
0x4D	Configuration MetaData			Adaptive NVM Configuration Meta Data for each feature.
0x50	5th Free Provisioning Area			Free area to be used as the new bank when updating future 256 KB long modules mapped outside the shadow RAM.
0x4E	Immediate Value			Adaptive NVM Immediate Value. Holds the currently used values of the immediate fields. This section keeps copies of the values (shadow RAM)

6.1.2 NVM Header

Table 6-2 lists the format of the NVM header section. It includes words with a specific content and pointers to the first level of NVM modules. See Section 6.2 for more details.

- **Bold** items are pointers to modules.
- The Auth / RO column indicates the following:
 - Auth - The pointed module is authenticated and the pointer is read-only
 - RO - The pointed module and/or the pointer are read-only, or the word is read-only
 - No - The word or the module is read/write

A read-only item can be written only when either in the blank Flash programming mode or through the flow described in Section 3.4.8.4.

- The CSR Format column indicates whether the pointed module uses the formats described in Section 6.1.3.
- The Module is in the Shadow RAM column indicates whether the module is mapped into the basic banks mirrored into the internal shadow RAM. If marked as No, the most significant bit of the pointer (bit 15) must be set to 1b to indicate that the pointer value (in bits 14:0) is expressed in 4 KB sector units. Otherwise, it is expressed in word units.
- Maximum Provisioned Size indicates the following:
 - Modules mapped in shadow RAM — How much has been accounted for the module in the shadow RAM, though the module can exceed this limit as long as the entire shadow RAM contents is not exceeding its size.
 - Modules mapped outside shadow RAM — This is the maximum size the module can take because it has to fit within the associated free provisioning area.
 - The maximum size provisioned for the NVM header itself is 256 words.



Table 6-2. NVM Header Map

Word Address	Pointed Module Name / Word Name	Auth / RO	Accessed by	Loading Trigger	CSR Format	Module is in Shadow RAM	Typical Size	Maximum Provisioned Size
0x00	NVM Control Word 1	RO	HWEMP	POR				
0x01	RO Commands Version	RO	EMP	EMPR				
0x02	Reserved							
0x03	PCIe Analog	Auth	HW	POR	No	No	6 KB	8 KB
0x04	PHY Analog	Auth	HW	POR	No	No	6 KB	8 KB
0x05	Expansion/Option ROM	Auth	BIOS/HW	PCIR ¹	No	No	496 KB	1160 KB
0x06	RO PCIR Registers Auto-load	RO	HWEMP	PCIR	Yes	Yes	64 bytes	128 bytes
0x07	Auto Generated Pointers	No	EMP/SW	POR	Yes	Yes	16 bytes	64 bytes
0x08	PCIR Registers Auto-load	No	HWEMP	PCIR	Yes	Yes	512 bytes	1024 bytes
0x09	EMP Global	RO ²	EMP	GLOBR	No	No	512 bytes	8 KB
0x0A	RO PCIe LCB	RO	HW	PERST	Yes	Yes	768 bytes	1024 bytes
0x0B	EMP Image	Auth	EMP	EMPR	No	No	1248 KB	1160 KB
0x0C	Reserved							
0x0D	Reserved							
0x0E	Manageability	RO ⁴	EMP	EMPR	No	No	1 KB	8 KB
0x0F	EMP Settings	RO ⁴	EMP	CORER ³	No	No	1 KB	8 KB
0x10	SW Compatibility Word 1	No	SW/BIOS	POR ²				
0x11	SW Compatibility Word 2	No	SW/BIOS	POR ²				
0x12	SW Compatibility Word 3	No	SW/BIOS	POR ²				
0x13	SW Compatibility Word 4	No	SW/BIOS	POR ²				
0x14	SW Compatibility Word 5	No	SW/BIOS	POR ²				
0x15	PBA Flag	No	SW/BIOS	POR ²				
0x16	PBA Block	No	SW/BIOS	POR ⁴	No	Yes	12 bytes	12 bytes
0x17	Boot Configuration	No	BIOS	POR ⁷	No	Yes	3072 bytes	3072 bytes
0x18	Dev Starter Revision	No	SW/BIOS/EMP	POR				
0x19	SW Reserved Word 2	No	SW/BIOS	POR ²				
0x1A	SW Reserved Word 3	No	SW/BIOS	POR ²				
0x1B	SW Reserved Word 4	No	SW/BIOS	POR ²				
0x1C	SW Reserved Word 5	No	SW/BIOS	POR ²				
0x1D	SW Reserved Word 6	No	SW/BIOS	POR ²				
0x1E	SW Reserved Word 7	No	SW/BIOS	POR ²				
0x1F	SW Reserved Word 8	No	SW/BIOS	POR ²				
0x20	SW Reserved Word 9	No	SW/BIOS	POR ²				
0x21	SW Reserved Word 10	No	SW/BIOS	POR ²				
0x22	SW Reserved Word 11	No	SW/BIOS	POR ²				
0x23	SW Reserved Word 12	No	SW/BIOS	POR ²				
0x24	SW Reserved Word 13	No	SW/BIOS	POR ²				



Table 6-2. NVM Header Map

Word Address	Pointed Module Name / Word Name	Auth / RO	Accessed by	Loading Trigger	CSR Format	Module is in Shadow RAM	Typical Size	Maximum Provisioned Size
0x25	SW Reserved Word 14	No	SW/BIOS	POR ²				
0x26	SW Reserved Word 15	No	SW/BIOS	POR ²				
0x27	SW Reserved Word 16 -	No	SW/BIOS	POR				
0x28	SW Reserved Word 17 - Permanent SAN MAC Address	No	SW/BIOS/EMP	CORER	No	Yes	98 bytes	128 bytes
0x29	SW Reserved Word 18 - Map version	No	SW/BIOS	POR ²				
0x2A	SW Reserved Word 19 - NVM Image Version	No	SW/BIOS	POR ²				
0x2B	SW Reserved Word 20 - NVM Structure Version	No	SW/BIOS	POR ²				
0x2C	SW Reserved Word 21	No	SW/BIOS	POR ²				
0x2D	SW Reserved Word 22 - EETRACK ID 1	No	SW/BIOS	POR ²				
0x2E	SW Reserved Word 23 - EETRACK ID 2	No	SW/BIOS	POR ²				
0x2F	VPD Area	RO ⁵	EMP/VPD SW	POR ⁷	No	Yes	256 bytes	1024 bytes
0x30	PXE Setup Options	No	BIOS	POR ⁷	No	Yes	34 bytes	64 bytes
0x31	PXE Configuration Customization Options	No	BIOS	POR ⁷	No	Yes	34 bytes	64 bytes
0x32	PXE Version	No	BIOS	POR ²				
0x33	IBA Capabilities	No	BIOS	POR ²				
0x34	SW Reserved Word 24	No	BIOS	POR ²				
0x35	SW Reserved Word 25	No	BIOS	POR ²				
0x36	iSCSI Option ROM Version	No	BIOS	POR ²				
0x37	VLAN Configuration Block Pointer	No	SW/BIOS	CORER ⁷	No	Yes	218 bytes	256 bytes
0x38	POR Registers Auto-load	No	HW	POR	Yes	Yes	1536 bytes	2048 bytes
0x39	Reserved							
0x3A	Reserved for EMPR Registers Auto-load	RO	HW	POR	Yes	Yes	0 bytes	32 v
0x3B	GLOBR Registers Auto-load	No	HW	GLOBR	Yes	Yes	800 bytes	1024 bytes
0x3C	CORER Registers Auto-load	No	HW	CORER	Yes	Yes	41300 bytes	45056 bytes
0x3D	PHY configuration scripts	RO	EMP	POR	No	No		
0x3E	PCIe ALT Auto-load	RO ⁶	HW/EMP	PCIR ²	Yes	Yes	512 bytes	1KB
0x3F	SW Checksum	No	SW					
0x40	1st Free Provisioning Area	RO ⁴	EMP	EMPR ^{2,3}	No	No	1160 KB	1160 KB
0x41	1st Free Provisioning Area Size	RO	EMP	EMPR				
0x42	Reserved 4th Free Provisioning Area	RO ⁴	EMP	EMPR ^{2,3}	No	No	64 KB	64 KB
0x43	Reserved 4th Free Provisioning Area Size	RO	EMP	EMPR				



Table 6-2. NVM Header Map

Word Address	Pointed Module Name / Word Name	Auth / RO	Accessed by	Loading Trigger	CSR Format	Module is in Shadow RAM	Typical Size	Maximum Provisioned Size
0x44	3rd Free Provisioning Area	RO ⁴	EMP	EMPR ^{2,3}	No	No	16 KB	16 KB
0x45	3rd Free Provisioning Area Size	RO	EMP	EMPR				
0x46	2nd Free Provisioning Area	RO ⁴	EMP	EMPR ^{2,3}	No	No	8 KB	8 KB
0x47	2nd Free Provisioning Area Size	RO	EMP	EMPR				
0x48	EMP SR Settings	No	EMP	EMPR ⁷		Yes	82 bytes	128 bytes
0x49	Feature Configuration	RW	EMP	EMPR	No	Yes		
0x4A	Core Mem Config	RW	HW	POR	Yes	Yes	1 KB	1 KB
0x4C: 0x4B	Reserved	RO						
0x4D	Configuration Metadata	RO	EMP	EMPR	No	No	100 KB	128 KB
0x4E	Immediate Fields	RW	EMP	EMPR	No	Yes		
0x4F	External 25 GbE PHY Global	RO	EMP	EMPR	No	No	35 KB	128 KB
0x50	5th Free Provisioning Area	RO	EMP	EMPR ^{2,3}	No	No	256 KB	256 KB
0x51	5th Free Provisioning Area Size	RO	EMP	EMPR				
0x52	X710-TM4 Global Module	No	EMP	POR	No	No	209 KB	256 KB
0x59	SW-Data-Recovery	RO	EMP	PCIR ²	No	Yes	1 KB	2 KB
0x5A	PCIR-Data-Recovery	RO	EMP	PCIR ⁸	Yes	Yes	64 bytes	128 bytes

1. Only the pointer is loaded not the module's contents.
2. RO pointer, while the pointed area is RW.
3. Some items load on CORER, others on EMPR.
4. Pointer and module contents are loaded only into shadow RAM.
5. VPD pointer is a RO pointer, regardless to its own value and to the value of the GLPCI_CAPCTRL.VPD_EN bit. If the VPD *Write Enable* bit is cleared in the NVM, VPD area cannot be modified via the NVM Update admin command, but only via the VPD register set in the PF config space.
6. The module must be mapped to the shadow RAM word address 0x7E00. Any change performed to the module contents in shadow RAM is not dumped into Flash memory. Software cannot directly modify the module contents via NVM update commands, but only indirectly via the Alternate Structure admin commands.
7. Pointer and module contents.
8. Only used in recovery mode.

6.1.3 Structure of Hardware Modules

An NVM module read by hardware (as listed [Table 6-2](#)) is built according to one of the following structures:

- Fixed functionality structure – Each NVM word is allocated to one or more fixed fields and the contents of the NVM word are loaded to fixed CSRs in the X710-TM4. In [Table 6-2](#), these modules are referred as CSR format = No modules.
- Flexible functionality module – The structure defines the device address or addresses into which the NVM words are loaded. Therefore, the module might be used for different purposes based on its contents. In [Table 6-2](#), these modules are referred as CSR format = Yes modules.

The remainder of this section describes the structure of the contents of a flexible functionality module (excluding the module header described in [Section 6.1.5](#)). The general structure of the module is shown in [Figure 6-1](#).

A CSR format module can be made of a mix of repeating Type 1, 2, 3 and 4 segments. Each segment is described by its own header (the shaded fields in Figure 6-1).

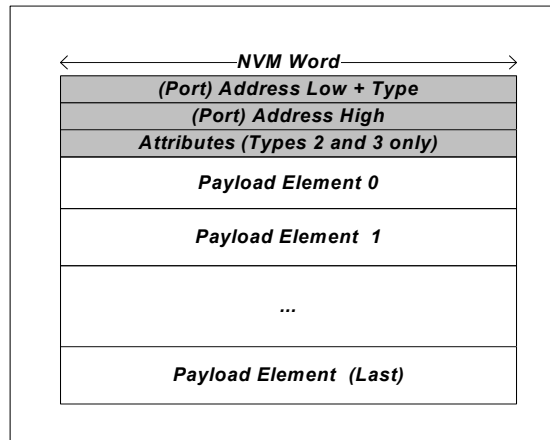


Figure 6-1. Structure of a Flexible Functionality Module

Description of the fields in Figure 6-1:

- **(Port) Address** — A 28-bit value that defines the starting address to which data Dwords are loaded. Serves as either a direct address to load into (Types 1 & 2) or as an indirect address referred to as a port address (Type 3 & 4). A port is made of a 32-bit address register followed by data registers of total size width. Address low is a 12-bit field mapped to word[15:4].
- **Type** — A 4-bit command field mapped to word[3:0] bits that defines the type of flexible module. Descriptions are as follows:
 - Type 1 (0001b) — Loads a single 32-bit data segment
 - Type 2 (0010b) — Load a set of 32-bit data Dwords into consecutive addresses
 - Type 3 (0011b) — Loads a set of 32-bit data Dwords through an address port
 - Type 4 (0100b) — Loads, through an address port, a sequence of data blocks into arbitrary addresses



- **Attributes** — A 16-bit optional word that defines attributes of the module
 - **Width** — A 3-bit field mapped to word[2:0] bits that describes the width (in 32-bit Dwords) of a data element
 - 000b = Data is 32-bit wide
 - 001b = Data is 64-bit wide
 - 010b = Data is 128-bit wide
 - 011b = Data is 256-bit wide
 - Else = Reserved
 - **Skip** — A 2-bit field mapped to word[4:3] bits that describes the number of address bytes to be skipped for getting the address of the next payload element (Type 2) or port register involved (Type 3 or 4).
 - 00b = Skip 4 bytes
 - 01b = Skip 8 x 4 bytes = 32 bytes
 - 10b = Skip 32 x 4 bytes = 128 bytes
 - 11b = Reserved
 - **Length** — An 11-bit field mapped to word[15:5] bits that contains the number of data elements, each of width bits.
- **Payload Elements** — A sequence of address or data elements to be loaded into the X710-TM4. The structure of the *Payload* field varies with each type and is described in the sections that follow.

6.1.3.1 Type 1 Module

Used to specify a single 32-bit data segment. The *Address* field defines the address into which the 32-bit data is loaded. See [Figure 6-2](#).

The common use of a Type 1 module is to concatenate a list of such structure to load a set of 32-bit values into various CSRs.

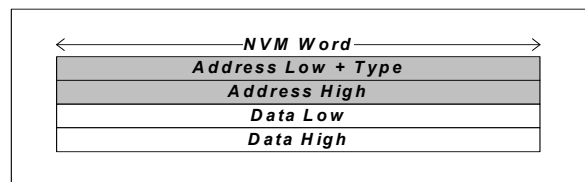


Figure 6-2. Structure of a Type 1 Module

6.1.3.2 Type 2 Module

Used to load a sequence of data into consecutive addresses, such as a memory array. See [Figure 6-3](#).

The *Address* field defines the first address to be loaded. Data is loaded as width-size elements into consecutive addresses. The *Attributes* field applies the following information:

- Width = 000b
- Skip = see previous description
- Length = see previous description

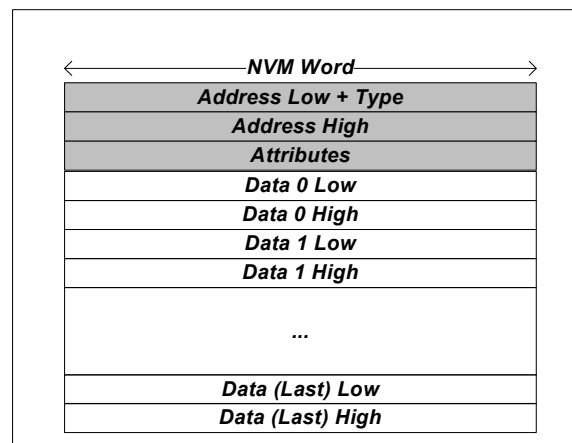


Figure 6-3. Structure of a Type 2 Module

6.1.3.3 Type 3 Module

Used to load a sequence of data through an address port.

[Figure 6-4](#) shows the structure of a Type 3 module. The following fields have special values:

- The *Port Address* field defines the address of the port address register through which address/data is loaded. The Port Address register is followed by the Port Data registers.
- The *Attributes* field applies with the following fields:
 - Width – The size of a single data element written into the port during one port load cycle
 - Length – Number of data elements written through the port
 - Skip – Encodes the number of bytes between the addresses of the port registers involved
- The *Indirect Address* field defines the first consecutive address to write to. It is written into the Port Address register for the first item and from then on the indirect address is increased (by the X710-TM4) with each write of a new data element from the list. Consecutive data elements are written into the port.
- The *Data Elements* field are made of length elements, each of the width Dwords. Each port load cycle handles one data element

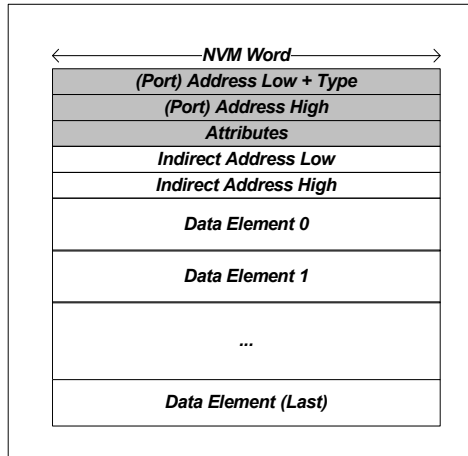
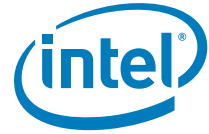


Figure 6-4. Structure of a Type 3 Module

The following figures describe two cases of using a Type 3 module:

- Figure 6-5 shows a case of loading 32-bit values into a Global (GL) port
- Figure 6-6 shows a case of loading 64-bit values into a PRT port

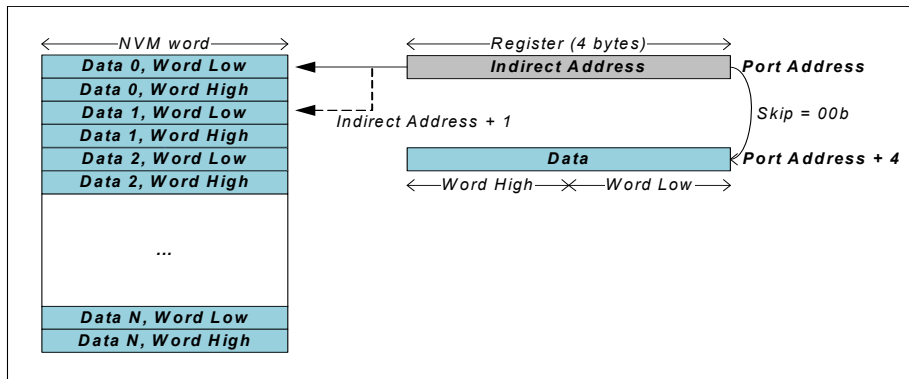


Figure 6-5. Access Through a Type 3 Module of 32-bits Wide Data Elements

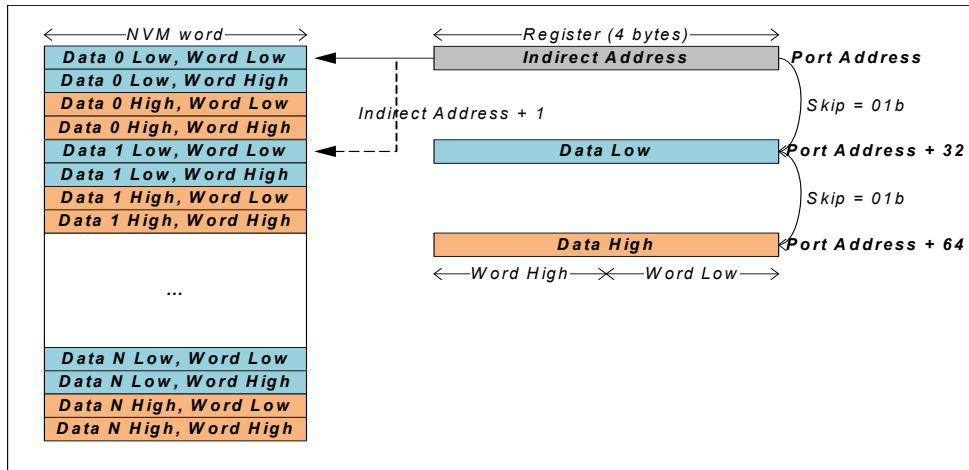


Figure 6-6. Access Through a Type 3 Module of 64-bits Wide Per Port Data Elements

6.1.3.4 Type 4 Module

Used to load, through an address port, a sequence of scattered data elements at arbitrary addresses.

Figure 6-7 shows the structure of a Type 4 module. The following fields have special values:

- The *Port Address* field defines the address of the port address register through which address/data is loaded. The port address register is followed by the port data registers.
- The *Attributes* field applies with the following fields:
 - *Width* — The size of a single data element written into the port during one port load cycle
 - *Length* — Number of data elements written through the port, not including the indirect address words (the words colored in gray in Figure 6-8 and Figure 6-9).
 - *Skip* — Encodes the number of bytes between the addresses of the port registers involved
- The *Indirect Address* fields define the address that the following data element is written into. It is written into the Port Address register.
- The *Data Element* is written into the port data register. The sequence of writing (the indirect address and its data element) into the port is repeated per each data element (length times).

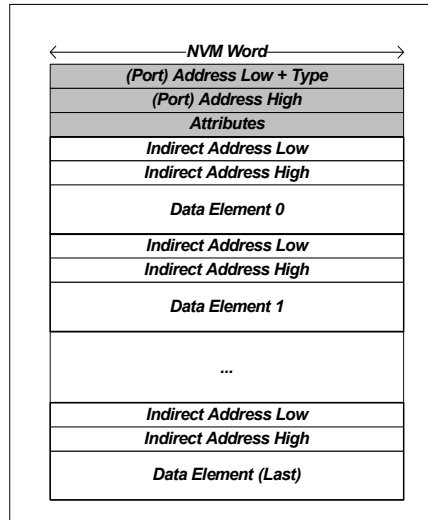


Figure 6-7. Structure of a Type 4 Module

The following figures show two cases of using a Type 4 module:

- Figure 6-8 shows a case of loading 32-bit values into a global (GL) port
- Figure 6-9 shows a case of loading 64-bit values into a PRT port

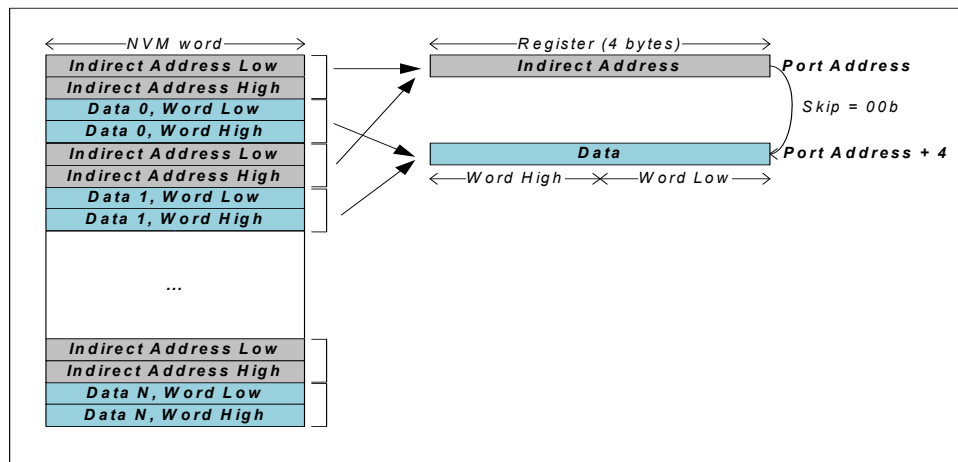


Figure 6-8. Access Through a Type 4 Module of 32-bits Wide Data Elements

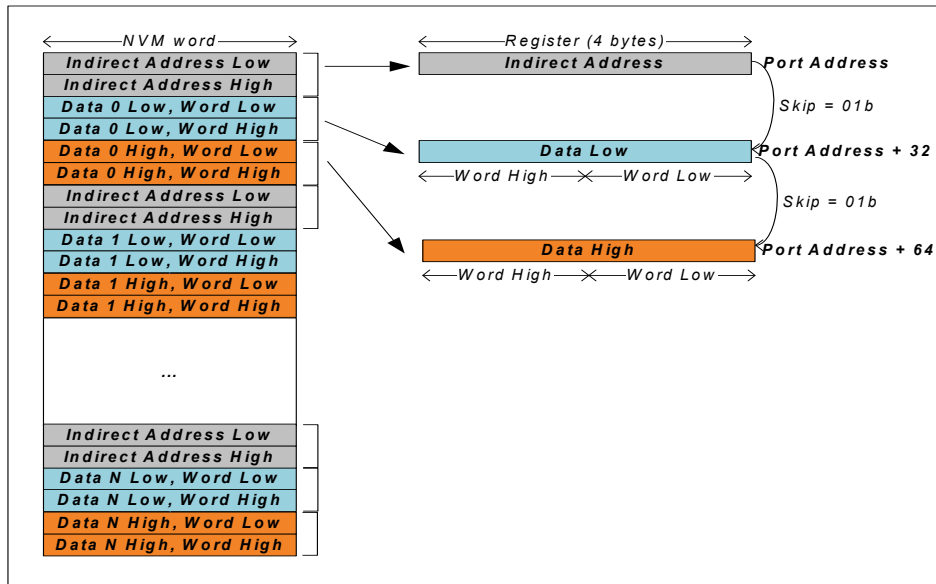


Figure 6-9. Access Through a Type 4 Module of 64-bits Wide Per Port Data Elements

6.1.3.5 Auto Generated Pointers

Type 1 and Type 2 items are generated by the auto-map generation tool from the project’s register database. These items are automatically mapped into the registers auto-load NVM modules according to their loading trigger (see Table 6-2) and with no possibility to perform manual changes. Until the list of Type 1/2 items stabilizes, each invocation of the tool might lead to a different mapping offset in the destination module.

EMP code and software tools cannot accommodate these variations in mapping offset, and hence, they need a fixed method for accessing some predefined Type 1/2 items. The auto-generated pointers module is automatically defined by the auto-map generation tool for this purpose. It is pointed to by NVM word 0x07. It lists the mapping address in NVM of the pre-defined Type 1/2 items via a 2-word structure per item.

The word address in shadow RAM of an item is given by the sum of the contents of its two associated words: pointer + offset. If the item is relative to an array of registers and/or to a register with a scope different than global, the offset is given for the first data word of the Type 1/2 array, for array instance [0] and scope instance [0].

In the auto-generated pointers module, the location of the 2-word structures relative to an item is made invariant along the project’s life. The alias name of the Type 1/2 register appears in the names of the two words in the structure.

Following is the list of auto-generated pointers, listed in the order by which they appear in the module:

1. PFPM_APM
2. PRTPM_GC
3. GLGEN_STAT



4. GLPCI_SERL
5. GLPCI_SERH
6. PRTGL_SAL
7. PRTGL_SAH
8. GLPCI_CAPSUP
9. PRTDCB_MFLCN
10. PRTDCB_FCCFG
11. PFGEN_PORTNUM
12. PFPCI_FUNC2
13. PFPCI_CLASS
14. PF_VT_PFALLOC
15. GLGEN_PCIFCNCNT
16. GLPCI_REVID
17. PFPCI_DEVID
18. GLPCI_SUBVENID
19. PFPCI_SUBSYSID
20. GLPCI_VENDORID
21. GLPCI_CNF2

6.1.4 NVM Integrity Checks by Software

This section describes the NVM integrity fields inserted in the the X710-TM4 NVM map to provide a basic detection of a faulty Flash part.

A software checksum check is performed by the PF driver just after completion of its initialization sequence. It covers only modules mapped into shadow RAM. If the checks fails, another try is attempted, and if it fails again, the PF driver disables Tx/Rx operations with the X710-TM4.

Modules mapped outside the shadow RAM include a CRC8 field. the EMP is responsible to check the CRC8 of the modules it loads/updates from/to Flash. It also provides software tools with the ability to check the integrity of these modules upon request because the PF driver does not include this check in its initialization sequence.

6.1.4.1 Software Checksum

The *Software Checksum* field covers the entire 64 KB shadow RAM contents (reserved words included) with the exception of the VPD area and to the PCIe ALT auto load module, which are skipped.

The *Software Checksum* word is located at word 0x3F. Its value is computed such that after adding all the covered words, including the *Software Checksum* word itself, the sum is 0xBABA.

Each time software tools are modifying one of these areas, it must update the *Software Checksum* field accordingly.

Each time the EMP is updating the contents of one of these areas by its own initiative (not as part of an NVM Update AQ command) or for handling an MC command received over SMBus or NC-SI, it must update the *Software Checksum* field accordingly.



6.1.4.2 CRC8

PCIe and PHY analog modules, and the EMP image have a separate CRC8 field embedded in their header. Refer to the *Module Format Version + CRC8* field in [Table 6-5](#) for its computing rules and coverage.

The CRC polynomial used is: X^8+X^2+X+1 .

Similarly, the following modules include CRC8 fields in their header and in the headers of their sub-modules:

- PE settings module
- EMP global module
- Manageability module
- EMP settings module

The EMP is responsible to check CRC8 validity of these modules (and their included sub-modules) on every EMPR, except to the PE settings module that is handled by PE on CORER events. If CRC8 is invalid after two tries (every try takes 260 ms for a 800 KB long EMP image), the EMP must report the error in the `GL_MNG_FWSM.EXT_ERR_IND` and `CRC_ERROR_MODULE` register fields by posting to the MC an AEN # 0x82 - NVM error (enable bit = 18) over NC-SI and by using the Status Data Byte 2[5] in the SMBus alert.

If the CRC8 error occurred on the EMP code, the ROM-EMP must report the error only to the host by setting a bit in a register and must not run the code loaded from RAM. It must also open the hardware NVM security to enable fixing the issue via software tools.

If the CRC8 error occurred on the manageability module, NC-SI/SMBus interfaces are disabled.

If the CRC8 error occurred on the EMP global module then the settings are not loaded to the PHY.

CRC8 of PE image and PE settings modules are checked by the PE itself. When an error occurs on the PE image, the PE processor is held in reset and the error is reported in the `GLPE_FWLDSTATUS` register. If the error occurs on PE settings, the settings are not used and the error is reported in the `GLPE_FWLDSTATUS` register.

Each time software tools are updating one of these modules, they must update its CRC8 field accordingly.

The EMP must check the concerned CRC8 fields before committing an NVM update command that was explicitly addressed for one of these modules. If a CRC8 field read from the free provisioning area of the Flash is not valid, the command completes with the EIO (Flash defect status).



6.1.4.3 NVM Integrity Summary

Table 6-3. NVM Integrity Summary Table

NVM Module	Integrity Check	When ¹	By	Number of Tries	Action on Error
Shadow RAM (excluding VPD area)	Checksum	Driver Init	PF driver	2	Disable Rx/Tx paths for the host (but not for MC) and report the failure to user/admin.
VPD area	No	N/A	N/A	N/A	N/A
PCIe Analog	CRC8	EMPR	EMP	2	Report the failure in GL_MNG_FWSM.EXT_ERR_IND. Post to the BMC an AEN # 0x82 - NVM error (enable bit = 18) over NC-SI and then use the Status Data Byte 2[5] in the SMBus alert.
PHY Analog	CRC8	EMPR	EMP	2	Report the failure in GL_MNG_FWSM.EXT_ERR_IND. Post to the BMC an AEN # 0x82 - NVM error (enable bit = 18) over NC-SI and then use the Status Data Byte 2[5] in the SMBus alert.
EMP Global	CRC8	EMPR	EMP	2	Report the failure in GL_MNG_FWSM.EXT_ERR_IND. Post to the BMC an AEN # 0x82 - NVM error (enable bit = 18) over NC-SI and then use the Status Data Byte 2[5] in the SMBus alert. Do not load the data into the PHY.
Manageability	CRC8	EMPR	EMP	2	Report the failure in GL_MNG_FWSM.EXT_ERR_IND. Disable NC-SI/SMBus.
EMP Settings	CRC8	EMPR	EMP	2	Report the failure in GL_MNG_FWSM.EXT_ERR_IND. Post to the BMC an AEN # 0x82 - NVM error (enable bit = 18) over NC-SI and then use the Status Data Byte 2[5] in the SMBus alert.
Option ROM	Yes	Boot Time	BIOS		
EMP Image	CRC8	EMPR	ROM-EMP	2	Report the failure in GL_MNG_FWSM.EXT_ERR_IND. Do not run the RAM code. Open hardware NVM security by clearing the FLA.LOCKED bit. Check CRC8 before committing an EMP image update. In case of an CRC8 error, report the error in the AQ completion and reject the update.

1. Besides the integrity check performed by the EMP before committing NVM update commands.



6.1.5 Header of NVM Modules

6.1.5.1 Header of all NVM Modules Mapped to Shadow RAM

Modules read by hardware do not contain pointers to sub-modules and they are not authenticated.

Table 6-4. NVM Header of Modules Mapped to Shadow RAM

Number of Words	Field or Segment Name	Description and Comments
1	Module Length	Length of the module contents expressed in words (<i>Module Length</i> field excluded). It must be set to N. Modules are size limited to the size of the shadow RAM (64 KB).
N	Word 1	
	Word 2	
	...	
	Word N	

6.1.5.2 Header/Trailer of Authenticated NVM Modules

This section describes the following modules:

1. EMP image (pointed by NVM word 0x0B).
2. PCIe analog (pointed by NVM word 0x03).
3. PHY analog (pointed by NVM word 0x04).
4. Option ROM (pointed by NVM word 0x05).

For option ROM, the first 330 words listed in [Table 6-5](#) are mapped at the end of the area allocated to the module (at its trailer), though for the sake of the authentication, these words are mapped at the module's header, as listed in [Table 6-5](#).

In [Table 6-5](#), fields colored in cyan are protected by the authentication signature.

Table 6-5. Header of Authenticated NVM Modules

Number of Words	Field or Segment Name	Description and Comments
64	CSS Header	Refer to Section 3.4.9.2 .
128	RSA Public Key	Refer to Section 3.4.9 . This field is skipped due to SHA256 Hash computing.
2	RSA Exponent	Refer to Section 3.4.9 . This field is skipped due to SHA256 Hash computing.
128	Encrypted SHA256 Hash	Refer to Section 3.4.9 . This field is skipped due to SHA256 Hash computing.
1	X710-TM4 Blank NVM Device ID	A unique Intel-provided device ID that identifies the X710-TM4 controller among other Intel controllers. It must be set to 0x154B in the X710-TM4.



Table 6-5. Header of Authenticated NVM Modules (Continued)

Number of Words	Field or Segment Name	Description and Comments
2	Max Module Area	It is the maximum Flash area expressed in words that can be used by the module, starting from CSS header (included). It is set to 580 K words (1160 KB) for an EMP image module and to 4 K words (8 KB) for other authenticated modules.
2	Current Module Area	It is the Flash area expressed in words that is currently used by the module, starting from CSS header (included).
1	Module Format Version + CRC8	Bit 15 = CRC8 field is used. Set to 1b if a CRC8 is computed over the module, set to 0b otherwise. It must always be set to 0b in the OROM module. Bits 14:8 = Module format version. Set to 0x02 to use the currently-defined format. Bits 7:0 = CRC8 value computed over the entire area allocated to the module (1160 KB for EMP image), starting from CSS header (excluding the fields not covered by the RSA authentication signature) and including all the remaining bytes of the area (excluding this word that is skipped for the sake of CRC8 computing).
1	Code Revision	Bits 15:8 = Major revision number. Bits 7:0 = Minor revision number. Image revision number (not necessarily referring to any embedded code).
1	Reserved Spare Word	Must be zeroed.
2 or 1	Parent Module Length	Length of the parent module contents expressed in words, module header and <i>Parent Module Length</i> field excluded. It excludes all the descendant modules. Modules read by firmware are NOT size limited to 128 KB. <ul style="list-style-type: none"> For modules parsed by firmware (EMP image), this length field is two words long and it covers for the firmware code, its padding words, and for the EMP image also the last 4 KB sector reserved for the RO commands section. It must be set to 0x0006DEB4. For modules parsed by hardware (PCIe analog and PHY analog modules), this length field is one word long and it does not include padded words.
N	Parent Word 1	Firmware module content formatting is specific to each module.
	Parent Word 2	
	...	
	Parent Word N	

6.1.6 Trailer of the EMP Image Module

In Table 6-6, fields colored in cyan are protected by the authentication signature.

The last 4 KB sector of the EMP image has the following format:



Table 6-6. RO Commands Section Format

Number of Words	Field or Segment Name	Description and Comments
1	RO Commands Version	Default is 0xFFFF, which means the section is empty and the remaining words are discarded.
1	X710-TM4 Blank NVM Device ID	A unique Intel-provided device ID that identifies the X710-TM4 controller among other Intel controllers. It must be set to 0x154B in the X710-TM4.
1	Minimum EMP Code Revision	Minimum EMP code revision number required for being able to parse the RO commands section. It must be lower or equal to the code revision number read from the EMP image header listed in Table 6-5.
1	RO Commands Length	Length in words (N), starting from next word.
N	RO Commands word 1	Format of the RO Commands is described in Section 6.1.6.1.
	RO Commands word 2	
	...	
	RO Commands word N	

6.1.6.1 Format of the RO Commands

The RO command words can contain the following two structures:

1. Shadow RAM Word Write command (2 words)
2. CSR Write command (4 words)

Each command starts with a type field.

Table 6-7 lists the different commands types:

Table 6-7. RO Commands Types

Type	Description
xxx1b	Word auto load.
0010b	CSR auto load.
Other	Invalid type, parsing is stopped here.

Table 6-8. Shadow RAM Word Write Command

Word Address [14:0] 15b	Type: 1b = 1'b1
Word Data [15:0] 16b	



Table 6-9. CSR Write Command

CSR Address [11:0] 12b	Type: 4b = 4'b0010
CSR Address [27:12] 16b	
CSR Data [15:0] 16b	
CSR Data [31:16] 16b	



NOTE: *This page intentionally left blank.*



6.2 NVM General Summary Table

Word Address	Used By	Word Name	Reference
0x0000	HW	NVM Control Word 1	Section 6.2.1.1
0x0001	FW	RO Commands Version	Section 6.2.1.2
0x0003	HW	PCIe Analog Module Pointer	Section 6.2.1.3
0x0004	HW	PHY Analog Module Pointer	Section 6.2.1.4
0x0005	HW	OROM Pointer	Section 6.2.1.5
0x0006	HW	RO PCIR Registers Auto-load Module Pointer	Section 6.2.1.6
0x0007	FW	Auto Generated Pointers Pointer	Section 6.2.1.7
0x0008	HW	PCIR Registers Auto-load Module Pointer	Section 6.2.1.8
0x0009	FW	EMP Global Module Pointer	Section 6.2.1.9
0x000A	HW	RO PCIe LCB Module Pointer	Section 6.2.1.10
0x000B	FW	EMP Image Pointer	Section 6.2.1.11
0x000D	HW	CSR Protected List Pointer	Section 6.2.1.12
0x000E	FW	Manageability Module Pointer	Section 6.2.1.13
0x000F	FW	EMP Settings Module Pointer	Section 6.2.1.14
0x0010	SW	SW Compatibility Word 1	Section 6.2.1.15
0x0011	SW	SW Compatibility Word 2	Section 6.2.1.16
0x0012	SW	SW Compatibility Word 3	Section 6.2.1.17
0x0013	SW	SW Compatibility Word 4	Section 6.2.1.18
0x0014	SW	SW Compatibility Word 5	Section 6.2.1.19
0x0015	SW	PBA Flags	Section 6.2.1.20
0x0016	SW	PBA Block Pointer	Section 6.2.1.21
0x0017	SW	Boot Configuration Start Address	Section 6.2.1.22
0x0018	SW	Software Reserved Word 1 - Dev Starter Version	Section 6.2.1.23
0x0019	SW	Software Reserved Word 2	Section 6.2.1.24
0x001A	SW	Software Reserved Word 3	Section 6.2.1.25
0x001B	SW	Software Reserved Word 4	Section 6.2.1.26
0x001C	SW	Software Reserved Word 5	Section 6.2.1.27
0x001D	SW	Software Reserved Word 6	Section 6.2.1.28
0x001E	SW	Software Reserved Word 7	Section 6.2.1.29
0x001F	SW	Software Reserved Word 8	Section 6.2.1.30
0x0020	SW	Software Reserved Word 9	Section 6.2.1.31
0x0021	SW	Software Reserved Word 10	Section 6.2.1.32
0x0022	SW	Software Reserved Word 11	Section 6.2.1.33
0x0023	SW	Software Reserved Word 12	Section 6.2.1.34
0x0024	SW	Software Reserved Word 13	Section 6.2.1.35
0x0025	SW	Software Reserved Word 14	Section 6.2.1.36
0x0026	SW	Software Reserved Word 15 - External Thermal Sensor	Section 6.2.1.37
0x0027	SW	Software Reserved Word 16	Section 6.2.1.38
0x0029	SW	Software Reserved Word 18 - Map Version	Section 6.2.1.39



Word Address	Used By	Word Name	Reference
0x002A	SW	Software Reserved Word 19 - NVM_Image_Version	Section 6.2.1.40
0x002B	SW	Software Reserved Word 20 - NVM Structure Version	Section 6.2.1.41
0x002D	SW	Software Reserved Word 22 - EETRACK ID 1	Section 6.2.1.42
0x002E	SW	Software Reserved Word 23 - EETRACK ID 2	Section 6.2.1.43
0x002F	SW	VPD Module Pointer	Section 6.2.1.44
0x0030	SW	PXE Setup Options Pointer	Section 6.2.1.45
0x0031	SW	PXE Configuration Customization Options Pointer	Section 6.2.1.46
0x0032	SW	PXE Version	Section 6.2.1.47
0x0033	SW	IBA Capabilities	Section 6.2.1.48
0x0034	SW	Software Reserved Word 24 - Original EETRACK ID 1	Section 6.2.1.49
0x0035	SW	Software Reserved Word 25 - Original EETRACK ID 2	Section 6.2.1.50
0x0036	SW	iSCSI Option ROM Version	Section 6.2.1.51
0x0037	SW	VLAN Configuration Block Pointer	Section 6.2.1.52
0x0038	HW	POR Registers Auto-load Module Pointer	Section 6.2.1.53
0x003A	HW	EMPR Registers Auto-load Pointer	Section 6.2.1.54
0x003B	HW	GLOBR Registers Auto-load Pointer	Section 6.2.1.55
0x003C	HW	CORER Registers Auto-load Pointer	Section 6.2.1.56
0x003F	SW	Software Checksum	Section 6.2.1.57
0x0040	FW	1st Free Provisioning Area Pointer	Section 6.2.1.58
0x0041	FW	1st Free Provisioning Area Size	Section 6.2.1.59
0x0042	FW	Reserved for 4th Free Provisioning Area Pointer	Section 6.2.1.60
0x0043	FW	Reserved for 4th Free Provisioning Area Size	Section 6.2.1.61
0x0044	FW	3rd Free Provisioning Area Pointer	Section 6.2.1.62
0x0045	FW	3rd Free Provisioning Area Size	Section 6.2.1.63
0x0046	FW	2nd Free Provisioning Area Pointer	Section 6.2.1.64
0x0047	FW	2nd Free Provisioning Area Size	Section 6.2.1.65
0x0048	HW	EMP SR Settings Pointer	Section 6.2.1.66
0x0049	HW	Feature Configuration Pointer	Section 6.2.1.67
0x004A	HW	Core Mem Config Pointer	Section 6.2.1.68
0x004D	HW	Configuration MetaData Pointer	Section 6.2.1.69
0x004E	HW	Immediate Values Pointer	Section 6.2.1.70
0x004F	FW	External PHY Global Module	Section 6.2.1.71
0x0050	HW	5th Free Provisioning Area	Section 6.2.1.72
0x0051	HW	5th Free Provisioning Area Size	Section 6.2.1.73
0x0052	HW	X710-TM4 Global Module	Section 6.2.1.74
0x0053	HW	Reserved	Section 6.2.1.75
0x0054	HW	Reserved	Section 6.2.1.76
0x0055	HW	Reserved	Section 6.2.1.77
0x0056	HW	Reserved	Section 6.2.1.78
0x0057	HW	Reserved	Section 6.2.1.79
0x0058	HW	Reserved	Section 6.2.1.80



Word Address	Used By	Word Name	Reference
0x0059	HW	SW-Data-Recovery	Section 6.2.1.81
0x005A	HW	PCIR-Data-Recovery	Section 6.2.1.82
0x005B + 1*n, n=0...166	HW	Spare NVM Header Words	Section 6.2.1.83

6.2.1 Init Module Section Summary Table

This NVM header module contains pointers to all other first-level sections. It includes the words that are relative to the entire NVM map.

For inner structure, see [Section 6.2](#).

6.2.1.1 NVM Control Word 1 - 0x0000

Bits	Field Name	Default NVM Value	Description
15:12	Reserved	0x0	Reserved.
7:6	NVM Validity	0x1	The <i>Signature</i> field indicates to the device that there is a valid NVM present. If the <i>Signature</i> field is not 01b, the other bits in this word are ignored, no further NVM read is performed, and the default values are used for the configuration space IDs. 0x0 = NVM not present 0. 0x1 = NVM present. 0x2 = NVM not present 2. 0x3 = NVM not present 3.
5	Reserved	0x0	Reserved.

6.2.1.2 RO Commands Version - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	RO Commands Version	0xFFFF	Contains the version of the RO commands section mapped to the last 4 KB sector of the EMP image.

6.2.1.3 PCIe Analog Module Pointer - 0x0003

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type: 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	PCIe Analog Configuration Module Pointer	0x0	

The PCIe Analog Configuration Module Pointer points to the PCIe Analog section. For PCIe Analog inner structure detail, see [Section 6.2.32](#).



6.2.1.4 PHY Analog Module Pointer - 0x0004

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	PHY Analog Module Pointer	0x0	

The PHY Analog Module Pointer points to the PHY Analog section. For PHY Analog inner structure detail, see [Section 6.2.33](#).

6.2.1.5 OROM Pointer - 0x0005

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Only the 4 KB sector unit is supported for this pointer. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	PCIe Expansion/ Option ROM Pointer	0x7FFF	

The PCIe Expansion/Option ROM Pointer points to the OROM section. For OROM inner structure detail, see [Section 6.2.46](#).

6.2.1.6 RO PCIR Registers Auto-load Module Pointer - 0x0006

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	PCIR Registers Auto-load Module Pointer	0x7FFF	

The PCIR Registers Auto-load Module Pointer points to the RO PCIR Registers Auto-load Module section. For RO PCIR Registers Auto-load Module inner structure detail, see [Section 6.2.2](#).



6.2.1.7 Auto Generated Pointers Pointer - 0x0007

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	Auto Generated Pointers Module Pointer	0x7FFF	

The Auto Generated Pointers Module Pointer points to the Auto Generated Pointers Module section. For Auto Generated Pointers Module inner structure detail, see [Section 6.2.31](#).

6.2.1.8 PCIR Registers Auto-load Module Pointer - 0x0008

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	PCIe TL Shared Module Pointer	0x7FFF	

The PCIe TL Shared Module Pointer points to the PCIR Registers Auto-load Module section. For PCIR Registers Auto-load Module inner structure detail, see [Section 6.2.17](#).

6.2.1.9 EMP Global Module Pointer - 0x0009

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	EMP Global Module Pointer	0x7FFF	

The EMP Global Module Pointer points to the EMP Global Module section. For EMP Global Module inner structure detail, see [Section 6.2.35](#).



6.2.1.10 RO PCIe LCB Module Pointer - 0x000A

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	RO PCIe LCB Module Pointer	0x7FFF	

The RO PCIe LCB Module Pointer points to the RO PCIe LCB Module section. For RO PCIe LCB Module inner structure detail, see [Section 6.2.3](#).

6.2.1.11 EMP Image Pointer - 0x000B

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	EMP Image Pointer	0x7FFF	

The EMP Image Pointer points to the EMP Image section. For EMP Image inner structure detail, see [Section 6.2.47](#).

6.2.1.12 CSR Protected List Pointer - 0x000D

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	CSR Protected List Pointer	0x7FFF	

The CSR Protected List Pointer points to the CSR Protected List section. For CSR Protected List inner structure detail, see [Section 6.2.4](#).



6.2.1.13 Manageability Module Pointer - 0x000E

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	Manageability Configuration Module Pointer	0x7FFF	

The Manageability Configuration Module Pointer points to the Manageability Module Header section. For Manageability Module Header inner structure detail, see [Section 6.2.36](#).

6.2.1.14 EMP Settings Module Pointer - 0x000F

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	EMP Module Pointer	0x7FFF	

The EMP Module Pointer points to the EMP Settings Module Header section. For EMP Settings Module Header inner structure detail, see [Section 6.2.43](#).

6.2.1.15 SW Compatibility Word 1 - 0x0010

Five words in the NVM image are reserved for compatibility information. New bits within these fields are defined as the need arises for determining software compatibility between various hardware revisions.

Bits	Field Name	Default NVM Value	Description
15:12	Reserved	0x0	Reserved.
11	LOM	0x0	Indicates whether the NVM attached to LAN silicon contains dedicated module for option ROM. Used by option ROM update applications. 0b = NIC (Attached flash contains module for option ROM). 1b = LOM (Attached flash has no module for option ROM). Note: This field is preserved by the Intel NVM update tool.
10	Server	0x1	Legacy, not currently used. 1b = Server. 0b = Client. Note: This field is preserved by the Intel NVM update tool.
9	Reserved	0x0	Reserved.
8	OEM/Retail	0x0	Legacy, not currently used. 1b = OEM. 0b = Retail. Note: This field is preserved by the Intel NVM update tool.
7:0	Reserved	0x00	Reserved.



6.2.1.16 SW Compatibility Word 2 - 0x0011

Five words in the NVM image are reserved for compatibility information. New bits within these fields are defined as the need arises for determining software compatibility between various hardware revisions.

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.17 SW Compatibility Word 3 - 0x0012

Five words in the NVM image are reserved for compatibility information. New bits within these fields are defined as the need arises for determining software compatibility between various hardware revisions.

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.18 SW Compatibility Word 4 - 0x0013

Five words in the NVM image are reserved for compatibility information. New bits within these fields are defined as the need arises for determining software compatibility between various hardware revisions.

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.19 SW Compatibility Word 5 - 0x0014

Five words in the NVM image are reserved for compatibility information. New bits within these fields are defined as the need arises for determining software compatibility between various hardware revisions.

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.



6.2.1.20 PBA Flags - 0x0015

Bits	Field Name	Default NVM Value	Description
15:0	PBA Flags	0xFAFA	A flag value of 0xFAFA indicates that the PBA is stored in a separate PBA block.

6.2.1.21 PBA Block Pointer - 0x0016

Bits	Field Name	Default NVM Value	Description
15:0	PBA Block Pointer		

6.2.1.22 Boot Configuration Start Address - 0x0017

Address of iSCSI Boot configuration module. This is a word pointer. The block length is embedded in the module.

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	iSCSI Boot Configuration Start Address	0x7FFF	This module is long by 1504 bytes and must be mapped in the first valid 4 KB sector of the Flash.

The iSCSI Boot Configuration Start Address points to the Boot Configuration Block section. For Boot Configuration Block inner structure detail, see [Section 6.2.15](#).

6.2.1.23 Software Reserved Word 1 - Dev Starter Version - 0x0018

Dev_Starter map version used to produce this image and must be filled out manually,

Bits	Field Name	Default NVM Value	Description
15:12	Major	0x7	NVM major version.
11:8	Decimal Point	0x0	Decimal point, used by automatic NVM reading tools. Must always be set to 0x0.
7:0	Minor	0x10	NVM minor version.



6.2.1.24 Software Reserved Word 2 - 0x0019

Bits	Field Name	Default NVM Value	Description
15:4	Reserved	0xFFFF	Reserved.
3	WoL Control Port 3	0x1	Wake On LAN (WoL) feature for port 3. 1b = Disabled or not supported. 0b = Supported and enabled. Valid values are: 0x0 = Enabled Supported and enabled. 0x1 = Disabled or not supported. Note: This field is preserved by the Intel NVM update tool.
2	WoL Control Port 2	0x1	WoL feature for port 2. 1b = Disabled or not supported. 0b = Supported and enabled. Valid values are: 0x0 = Enabled Supported and enabled. 0x1 = Disabled or not supported. Note: This field is preserved by the Intel NVM update tool.
1	WoL Control Port 1	0x1	WoL feature for port 1. 1b = Disabled or not supported. 0b = Supported and enabled. Valid values are: 0x0 = Enabled Supported and enabled. 0x1 = Disabled or not supported. Note: This field is preserved by the Intel NVM update tool.
0	WoL Control Port 0	0x1	WoL feature for port 0. 1b = Disabled or not supported. 0b = Supported and enabled. Valid values are: 0x0 = Enabled Supported and enabled. 0x1 = Disabled or not supported. Note: This field is preserved by the Intel NVM update tool.

6.2.1.25 Software Reserved Word 3 - 0x001A

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.26 Software Reserved Word 4 - 0x001B

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.27 Software Reserved Word 5 - 0x001C

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.



6.2.1.28 Software Reserved Word 6 - 0x001D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.29 Software Reserved Word 7 - 0x001E

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.30 Software Reserved Word 8 - 0x001F

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.31 Software Reserved Word 9 - 0x0020

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.32 Software Reserved Word 10 - 0x0021

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.33 Software Reserved Word 11 - 0x0022

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.34 Software Reserved Word 12 - 0x0023

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.35 Software Reserved Word 13 - 0x0024

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.



6.2.1.36 Software Reserved Word 14 - 0x0025

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.37 Software Reserved Word 15 - External Thermal Sensor - 0x0026

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.

The External Thermal Sensor points to External Thermal Sensor Configuration Block section. For External Thermal Sensor Configuration Block inner structure detail, see [Section 6.2.40](#).

6.2.1.38 Software Reserved Word 16 - 0x0027

Bits	Field Name	Default NVM Value	Description
15:0	Reserved		Reserved.

6.2.1.39 Software Reserved Word 18 - Map Version - 0x0029

Automatically generated by the Intel NVM update tool.

Bits	Field Name	Default NVM Value	Description
15:0	Map Version		

6.2.1.40 Software Reserved Word 19 - NVM_Image_Version - 0x002A

Automatically generated by the Intel NVM update tool.

Bits	Field Name	Default NVM Value	Description
15:0	NVM Image Version		



6.2.1.41 Software Reserved Word 20 - NVM Structure Version - 0x002B

Bits	Field Name	Default NVM Value	Description
15:12	Major	0x4	NVM Major Version.
11:8	Decimal Point	0x0	Decimal point, used by automatic NVM reading tools. Must always be set to 0x0.
7:0	Minor	0x01	NVM Structure Minor Version.

6.2.1.42 Software Reserved Word 22 - EETRACK ID 1 - 0x002D

This word is for the first word of the eTrack_ID number written by the EEPROM manager tool.

Bits	Field Name	Default NVM Value	Description
15:0	eTrack_ID Word 1	0xFFFF	The EEPROM manager tool writes a unique 32-bit eTrack_ID number in two sequential NVM words. The eTrack_ID is written when the EEPROM manager tool creates an image on the Intel network. The eTrack_ID DB tracks NVM images back to a specific SCM build.

6.2.1.43 Software Reserved Word 23 - EETRACK ID 2 - 0x002E

This word is the second word of the eTrack_ID number written by the EEPROM manager tool.

Bits	Field Name	Default NVM Value	Description
15:0	eTrack_ID Word 2		

6.2.1.44 VPD Module Pointer - 0x002F

Word pointer to the VPD module. Block length is embedded in the module.

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	VPD Pointer	0x7FFF	Vital Product Data Pointer. 0x7FFF Default unless VPD relative section is specified. The VPD section size is usually 64 words and is initialized to 0 or 0x7FFF. During run time this module is accessible through the VPD capability in the PCI configuration space. This module must be mapped in the first valid 4 KB sector of the Flash.

The VPD Pointer points to the VPD Module section. For VPD Module inner structure detail, see [Section 6.2.7](#).



6.2.1.45 PXE Setup Options Pointer - 0x0030

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	PXE Setup Options Pointer	0x7FFF	

The PXE Setup Options Pointer points to PXE Setup Options section. For PXE Setup Options inner structure detail, see [Section 6.2.13](#).



6.2.1.46 PXE Configuration Customization Options Pointer - 0x0031

Word 0x31 of the NVM contains settings that can be programmed by an OEM or network administrator to customize the operation of software. These settings cannot be changed from within the Control- S setup menu. The lower byte contains settings that would typically be configured by a network administrator using an external utility; these settings generally control which setup menu options are changeable. The upper byte is generally settings that would be used by an OEM to control the operation of the agent in a LOM environment, although there is nothing in the agent to prevent their use on a NIC implementation. The default value for this word is 0x4000.

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	PXE Configuration Customization Options Pointer	0x7FFF	

The PXE Configuration Customization Options Pointer points to the PXE Configuration Customizations Options section. For PXE Configuration Customizations Options inner structure detail, see [Section 6.2.14](#).

6.2.1.47 PXE Version - 0x0032

Word 0x32 of the NVM is used to store the version of the boot agent that is stored in the Flash image. When the boot agent loads, it can check this value to determine if any first-time configuration needs to be performed. The boot agent then updates this word with its version. Some diagnostic tools to report the version of the boot agent in the Flash also read this word.

Bits	Field Name	Default NVM Value	Description
15:12	Major Version	0x0	PXE Boot Agent Major Version. Default value is 0x0. Note: This field is preserved by the Intel NVM update tool.
11:8	Minor Version	0x0	PXE Boot Agent Minor Version. Default value is 0x0. Note: This field is preserved by the Intel NVM update tool.
7:0	Build Number	0x0	PXE Boot Agent Build Number. Default value is 0x0. Note: This field is preserved by the Intel NVM update tool.

6.2.1.48 IBA Capabilities - 0x0033

Word 0x33 of the NVM is used to enumerate the boot technologies that have been programmed into the Flash. This is updated by Flash configuration tools and is not updated or read by the IBA.

Bits	Field Name	Default NVM Value	Description
15:14	Signature	0x1	Signature. Must be set to 01b to indicate that this word has been programmed by the boot agent or other configuration software. Note: This field is preserved by the Intel NVM update tool.



Bits	Field Name	Default NVM Value	Description
13	Allow PXE Disable	0x0	If set to 0b, PXE is always load regardless the settings in bits 2:0 in Main Setup Options word. If set to 1b, PXE is loaded when bits 2:0 in Main Setup Options are set to 0x0 (PXE enabled on this port). 0x0 = Ignore PXE disable. 0x1 = Allow PXE disable. Note: This field is preserved by the Intel NVM update tool.
12:6	Reserved	0x000	Reserved. Must be 0x0.
4	iSCSI boot	0x0	iSCSI is present if set to 1b. 0x0 = Not present. 0x1 = Present. Note: This field is preserved by the Intel NVM update tool.
3	efi ebd driver	0x0	EFI UNDI driver is present if set to 1b. 0x0 = Not present. 0x1 = Present. Note: This field is preserved by the Intel NVM update tool.
2	RPL	0x0	RPL module is present if set to 1b. Reserved bit for devices. 0x0 = Not present. 0x1 = Present. Note: This field is preserved by the Intel NVM update tool.
1	PXE/UNDI Driver	0x1	PXE UNDI driver is present if set to 1b. Valid values are: 0x0 = Not present 0x1 = Present Note: This field is preserved by the Intel NVM update tool.
0	PXE Base Code	0x1	PXE base code is present if set to 1b. 0x0 = Not present. 0x1 = Present. Note: This field is preserved by the Intel NVM update tool.



6.2.1.49 Software Reserved Word 24 - Original EETRACK ID 1 - 0x0034

Bits	Field Name	Default NVM Value	Description
15:0	Original EETRACK ID 1	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.1.50 Software Reserved Word 25 - Original EETRACK ID 2 - 0x0035

Bits	Field Name	Default NVM Value	Description
15:0	Original EETRACK ID 2	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.1.51 iSCSI Option ROM Version - 0x0036

Word 0x36 of the NVM is used to store the version of iSCSI Option ROM updated. The value must be above 0x2000. The value below (word 0x1FFF = 16 KB NVM size) is reserved for future expansion for a pointer to a combo option ROM component version structure. iSCSIUtil, FLAUtil, DMiX update iSCSI Option ROM versions if the value is above 0x2000, 0x0000, or 0xFFFF. The pointer (0x0040 - 0x1FFF) should be kept and not be overwritten.

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Note: This field is preserved by the Intel NVM update tool.

6.2.1.52 VLAN Configuration Block Pointer - 0x0037

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	VLAN Configuration Block Pointer	0x7FFF	

The VLAN Configuration Block Pointer points to the VLAN Configuration Block section. For VLAN Configuration Block inner structure detail, see [Section 6.2.16](#).



6.2.1.53 POR Registers Auto-load Module Pointer - 0x0038

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	POR Registers Auto-Load Module Pointer	0x7FFF	

The POR Registers Auto-load Module Pointer points to the POR Registers Auto-load Module section. For POR Registers Auto-load Module inner structure detail, see [Section 6.2.19](#).

6.2.1.54 EMPR Registers Auto-load Pointer - 0x003A

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	EMPR Registers Auto-Load Module Pointer	0x7FFF	

The EMPR Registers Auto-load Module Pointer points to the EMPR Auto-load section. For EMPR Auto-load inner structure detail, see [Section 6.2.5](#).

6.2.1.55 GLOBR Registers Auto-load Pointer - 0x003B

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	GLOBR Registers Auto-Load Module Pointer	0x7FFF	

The GLOBR Registers Auto-load Module Pointer points to the GLOBR Registers Auto-load Module section. For GLOBR Registers Auto-load Module inner structure detail, see [Section 6.2.21](#).



6.2.1.56 CORER Registers Auto-load Pointer - 0x003C

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	CORER Registers Auto-Load Module Pointer	0x7FFF	

The CORER Registers Auto-load Module Pointer points to the CORER Registers Auto-load Module section. For CORER Registers Auto-load Module inner structure detail, see [Section 6.2.20](#).

6.2.1.57 Software Checksum - 0x003F

Bits	Field Name	Default NVM Value	Description
15:0	Checksum		The software checksum field covers the entire 64 KB shadow RAM contents (reserved words included), except to the VPD area which is skipped. Its value is computed such that after adding all the covered words, including the software checksum word itself, the sum is 0xBABA. The checksum word is used to ensure that the base NVM image is a valid image. The initial value in the 16-bit summing register should be 0x0000 and the carry bit should be ignored after each addition. This word is used strictly by software. Hardware does not calculate nor check its content but rather checks the NVM validity field in the NVM Control Word 1.

6.2.1.58 1st Free Provisioning Area Pointer - 0x0040

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	1st Free Provisioning Area Pointer	0x7FFF	

The 1st Free Provisioning Area Pointer points to the 1st Free Provisioning Area section. For 1st Free Provisioning Area inner structure detail, see [Section 6.2.48](#).



6.2.1.59 1st Free Provisioning Area Size - 0x0041

Bits	Field Name	Default NVM Value	Description
15:10	Reserved	0x00	Reserved.
9:0	1st Free Provisioning Area Size	0x122	Size expressed in 4 KB sector units.

6.2.1.60 Reserved for 4th Free Provisioning Area Pointer - 0x0042

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	2nd Free Provisioning Area Pointer	0x7FFF	

The 2nd Free Provisioning Area Pointer points to the 4th Free Provisioning Area section. For 4th Free Provisioning Area inner structure detail, see [Section 6.2.49](#).

6.2.1.61 Reserved for 4th Free Provisioning Area Size - 0x0043

Bits	Field Name	Default NVM Value	Description
15:10	Reserved	0x00	Reserved.
9:0	2nd Free Provisioning Area Size	0x010	Size expressed in 4 KB sector units.

6.2.1.62 3rd Free Provisioning Area Pointer - 0x0044

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	3rd Free Provisioning Area Pointer	0x7FFF	

The 3rd Free Provisioning Area Pointer points to the 3rd Free Provisioning Area section. For 3rd Free Provisioning Area inner structure detail, see [Section 6.2.50](#).



6.2.1.63 3rd Free Provisioning Area Size - 0x0045

Bits	Field Name	Default NVM Value	Description
15:10	Reserved	0x00	Reserved.
9:0	3rd Free Provisioning Area Size	0x020	Size expressed in 4 KB sector units.

6.2.1.64 2nd Free Provisioning Area Pointer - 0x0046

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	4th Free Provisioning Area Pointer	0x7FFF	

The 4th Free Provisioning Area Pointer points to the 2nd Free Provisioning Area section. For 2nd Free Provisioning Area inner structure detail, see [Section 6.2.45](#).

6.2.1.65 2nd Free Provisioning Area Size - 0x0047

Bits	Field Name	Default NVM Value	Description
15:10	Reserved	0x00	Reserved.
9:0	4th Free Provisioning Area Size	0x002	Size expressed in 4 KB sector units.

6.2.1.66 EMP SR Settings Pointer - 0x0048

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	EMP SR Settings Pointer	0x7FFF	

The EMP SR Settings Pointer points to the EMP SR Settings Module Header section. For EMP SR Settings Module Header inner structure detail, see [Section 6.2.22](#).



6.2.1.67 Feature Configuration Pointer - 0x0049

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	Feature Configuration Pointer	0x7FFF	

The Feature Configuration Pointer points to the Feature Configuration section. For Feature Configuration inner structure detail, see [Section 6.2.27](#).

6.2.1.68 Core Mem Config Pointer - 0x004A

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	Core Mem Config Pointer	0x7FFF	

The Core Mem Config Pointer points to the Core Mem Config section. For Core Mem Config inner structure detail, see [Section 6.2.6](#).

6.2.1.69 Configuration MetaData Pointer - 0x004D

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	Configuration MetaData Pointer	0x7FFF	

The Configuration MetaData Pointer points to the Configuration MetaData - Dummy section. For Configuration MetaData - Dummy inner structure detail, see [Section 6.2.51](#).



6.2.1.70 Immediate Values Pointer - 0x004E

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x0	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	Immediate Values Pointer	0x7FFF	

The Immediate Values Pointer points to the Immediate Values section. For Immediate Values inner structure detail, see [Section 6.2.28](#).

6.2.1.71 External PHY Global Module - 0x004F

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	Immediate Values Pointer	0x7FFF	

The Immediate Values Pointer points to the External PHY Global Module section. For External PHY Global Module inner structure detail, see [Section 6.2.53](#).

6.2.1.72 5th Free Provisioning Area - 0x0050

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	5th Free Provisioning Area Pointer	0x7FFF	

The 5th Free Provisioning Area Pointer points to the 5th Free Provisioning Area section. For 5th Free Provisioning Area inner structure detail, see [Section 6.2.57](#).



6.2.1.73 5th Free Provisioning Area Size - 0x0051

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	Reserved.
14:0	5th Free Provisioning Area Size	0x040	Size expressed in 4 KB sector units.

6.2.1.74 X710-TM4 Global Module - 0x0052

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0x1	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	X710-TM4 Global Module Pointer	0x7FFF	

The X710-TM4 Global Module Pointer points to the X710-TM4 Global Module section. For the X710-TM4 Global Module inner structure detail, see [Section 6.2.58](#).



6.2.1.75 Reserved - 0x0053

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.76 Reserved - 0x0054

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.77 Reserved - 0x0055

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.78 Reserved - 0x0056

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.79 Reserved - 0x0057

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.1.80 Reserved - 0x0058

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.



6.2.1.81 SW-Data-Recovery - 0x0059

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	1b	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	SW-Data-Recovery Pointer	0x7FFF	

The SW-Data-Recovery Pointer points to the SW-Data-Recovery section. For SW-Data-Recovery Section inner structure detail, see [Section 6.2.4](#).

6.2.1.82 PCIR-Data-Recovery 0x005A)

Bits	Field Name	Default NVM Value	Description
15	Pointer Type	0b	Pointer Type. 0b = Word units. 1b = 4 KB sector units. Valid values are: 0x0 = Word units. 0x1 = 4 KB sector units.
14:0	SW-Data-Recovery Pointer	0x7FFF	

The PCIR-Data-Recovery Pointer points to the PCIR-Data-Recovery section. For PCIR-Data-Recovery Section inner structure detail, see [Section 6.2.9](#).

6.2.1.83 Spare NVM Header Words[n] (0x005D + 1*n, n=0...166)

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	Reserved.

6.2.2 RO PCIR Registers Auto-load Module Section Summary Table

Contains RO parameters that configure the PCIe transaction layer.

Word Offset	Description	Reference
0x0000	Module Length	See section 6.2.2.1
0x0001	LO PCIR Data	See section 6.2.2.2



6.2.2.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		

6.2.2.2 LO PCIR Data - 0x0001

Raw data module length: variable

This word must be disabled at the image level.

6.2.3 RO PCIe LCB Module Section Summary Table

Contains RO parameters that configure the PCIe link layer (LCB unit).

Word Offset	Description	Reference
0x0000	Module Length	See section 6.2.3.1
0x0001	RO PCIe LCB Data	See section 6.2.3.2

6.2.3.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		

6.2.3.2 RO PCIe LCB Data - 0x0001

Raw data module length: variable

This word must be disabled at the image level.

6.2.4 CSR Protected List Section Summary Table

Defines the list of the protected CSRs and their default settings. These registers are made RO to the host as they contain settings that are critical for the host to Flash access when in blank Flash programming mode.

Word Offset	Description	Reference
0x0000	Module length	See section 6.2.4.1
0x0001	Reserved	See section 6.2.4.2
0x0002 - 0006	NVM contents for GLPCI_LCBADD	See section 6.2.4.3
0x0007 - 0008	NVM contents for GLPCI_LCBDATA	See section 6.2.4.4
0x0009 - 000D	NVM contents for PRTMAC_PHY_ANA_ADD	See section 6.2.4.5
0x0010 - 0014	NVM contents for MEM_INIT_GATE_AL_STR	See section 6.2.4.6
0x0017 - 001A	NVM contents for GLGEN_STAT	See section 6.2.4.7



Word Offset	Description	Reference
0x0027 - 002B	NVM contents for GLPCI_ANA_ADD	See section 6.2.4.8
0x002E - 0031	NVM contents for GLPCI_LBARCTRL	See section 6.2.4.9
0x0032 - 00AC	NVM contents for GLNVM_PROTCSR	See section 6.2.4.10

6.2.4.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		Length in: 2 bytes unit - 1 First Section -> Word: CSR Protected List -> Module Length Last Section -> Word: CSR Protected List -> Starting Address Low at GLNVM_PROTCSR[0]

6.2.4.2 Reserved - 0x0001

This word must be disabled at the image level.

Bits	Field Name	Default NVM Value	Description
15:0	Reserved		Reserved.

6.2.4.3 GLPCI_LCBADD - (0x0002 - 0006)

6.2.4.3.1 Starting Address Low at GLPCI_LCBADD - 0x0002

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPCI_LCBADD	0x9C4C0	
3:0	Type	0x2	

6.2.4.3.2 Starting Address High at GLPCI_LCBADD - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPCI_LCBADD		



6.2.4.3.3 Attributes at GLPCI_LCBADD - 0x0004

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.4.3.4 Data Low of GLPCI_LCBADD - 0x0005

6.2.4.3.5 Data High of GLPCI_LCBADD - 0x0006

6.2.4.4 GLPCI_LCBDATA - (0x0007 - 0008)

6.2.4.4.1 Data Low of GLPCI_LCBDATA - 0x0007

6.2.4.4.2 Data High of GLPCI_LCBDATA - 0x0008

6.2.4.5 PRTMAC_PHY_ANA_ADD - (0x0009 - 000D)

6.2.4.5.1 Starting Address Low at PRTMAC_PHY_ANA_ADD - 0x0009

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_PHY_ANA_ADD	0xA4038	
3:0	Type	0x2	

6.2.4.5.2 Starting Address High at PRTMAC_PHY_ANA_ADD - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_PHY_ANA_ADD		

6.2.4.5.3 Attributes at PRTMAC_PHY_ANA_ADD - 0x000B

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.4.6 MEM_INIT_GATE_AL_STR - (0x0010 - 0014)

6.2.4.6.1 Starting Address Low at MEM_INIT_GATE_AL_STR - 0x0010

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of MEM_INIT_GATE_AL_STR	0xB6000	
3:0	Type	0x2	

6.2.4.6.2 Starting Address High at MEM_INIT_GATE_AL_STR - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of MEM_INIT_GATE_AL_STR		

6.2.4.6.3 Attributes at MEM_INIT_GATE_AL_STR - 0x0012

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.4.7 GLGEN_STAT - (0x0017 - 001A)

6.2.4.7.1 Address Low at GLGEN_STAT - 0x0017

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLGEN_STAT	0xB612C	
3:0	Type	0x1	

6.2.4.7.2 Address High at GLGEN_STAT - 0x0018

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLGEN_STAT		

6.2.4.7.3 Data Low of GLGEN_STAT - 0x0019

6.2.4.7.4 Data High of GLGEN_STAT - 0x001A



6.2.4.8 GLPCI_ANA_ADD - (0x0027 - 002B)

6.2.4.8.1 Starting Address Low at GLPCI_ANA_ADD - 0x0027

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPCI_ANA_ADD	0xBA000	
3:0	Type	0x2	

6.2.4.8.2 Starting Address High at GLPCI_ANA_ADD - 0x0028

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPCI_ANA_ADD		

6.2.4.8.3 Attributes at GLPCI_ANA_ADD - 0x0029

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.4.9 GLPCI_LBARCTRL - (0x002E - 0031)

6.2.4.9.1 Address Low at GLPCI_LBARCTRL - 0x002E

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPCI_LBARCTRL	0xBE484	
3:0	Type	0x1	

6.2.4.9.2 Address High at GLPCI_LBARCTRL - 0x002F

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPCI_LBARCTRL		

6.2.4.9.3 Data Low of GLPCI_LBARCTRL - 0x0030

6.2.4.9.4 Data High of GLPCI_LBARCTRL - 0x0031



6.2.4.10 GLNVM_PROTCSR - (0x0032 - 00AC)

6.2.4.10.1 Starting Address Low at GLNVM_PROTCSR - 0x0032

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLNVM_PROTCSR	0xB6010	
3:0	Type	0x2	

6.2.4.10.2 Starting Address High at GLNVM_PROTCSR - 0x0033

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLNVM_PROTCSR		

6.2.4.10.3 Attributes at GLNVM_PROTCSR - 0x0034

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x3C	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.4.10.4 Data Low of GLNVM_PROTCSR[n] - 0x0035 + 2*n, n=0...59

6.2.4.10.5 Data High of GLNVM_PROTCSR[n] - 0x0036 + 2*n, n=0...59

6.2.5 EMPR Auto-load Section Summary Table

Read margin settings to the device’s core memories that are loaded by hardware auto-load only on the first EMPR event that occurs after POR.

The contents of this module is checked against the CSR protected list.

Word Offset	Description	Reference
0x0000	Module Length	See section 6.2.5.1
0x0001	EMPR Auto-load Data	See section 6.2.5.2



6.2.5.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		Length in: 2 bytes unit - 1 First Section -> Word: EMPR Auto-load -> Module Length Last Section -> Word: EMPR Auto-load -> EMPR Auto-load Data

6.2.5.2 EMPR Auto-load Data - 0x0001

Raw data module length: variable

Contains patches

6.2.6 Core Mem Config Section Summary Table

Read margin settings to the device’s core memories that are loaded by hardware auto-load on every CORER events.

The contents of this module is checked against the CSR Protected List.

Word Offset	Description	Reference
0x0000	Module Length	See section 6.2.6.1
0x0001	Core Mem Config Data	See section 6.2.6.2

6.2.6.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		Length in: 2 bytes unit - 1 First Section -> Word: Core Mem Config -> Module Length Last Section -> Word: Core Mem Config -> Core Mem Config Data

6.2.6.2 Core Mem Config Data - 0x0001

Raw data module length: variable

6.2.7 VPD Module Section Summary Table

VPD loaded by an OEM. It contains RO and RW information about the NIC/LOM.

Word Offset	Description	Reference
0x0000	[New Word]	See section 6.2.7.1

6.2.7.1 [New Word] - 0x0000

Raw data module length: 512 words

Note: This word is preserved by the Intel NVM update tool.



6.2.8 SW-Data-Recovery Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	Section 6.2.8.1
0x0001	Section Header	Section 6.2.8.2
0x0002	Original EE-Track-ID Word 1	Section 6.2.8.3
0x0003	Original EE-Track-ID Word 2	Section 6.2.8.4
0x0004 + 1*n, n=0...4	PBA	Section 6.2.8.5
0x0009	MAC Addresses Length	Section 6.2.8.6
0x000A + 3*n, n=0...15	MAC Addresses Word 0	Section 6.2.8.7
0x000B + 3*n, n=0...15	MAC Addresses Word 1	Section 6.2.8.8
0x000C + 3*n, n=0...15	MAC Addresses Word 2	Section 6.2.8.9

6.2.8.1 Section Length - 0x0000

Bits	Field Name	NVM Image Value	Description
15:0	Section Length	<Length>	Length in: 2 bytes unit - 1 First Section -> Word: SW-Data-Recovery Section -> Section Length Last Section -> Word: SW-Data-Recovery Section -> MAC Addresses Word 2

6.2.8.2 Section Header - 0x0001

- [0] = Device ID field is valid (devid_valid)
- [1] = EETRACK-ID field is valid (eetid_valid).
- [2] = PBA field is valid (pba_valid).
- [3] = MAC addresses field is valid (sa_valid).
- [Other] = Reserved.



Bits	Field Name	NVM Image Value	Description
15	Recovery Mode	0x0	Define the current status of NVM-Recovery-Mode. Set by firmware when entering recovery mode to keep the status POR-persistent. Firmware must clear this bit after an NVM update is done by software.
14:4	Reserved	0x0	Reserved.
3	MAC Addresses Valid	0x0	Define the validity of the data value of the MAC addresses in the recovery section: 0b = Data wasn't programmed by firmware. 1b = Data programmed and valid.
2	PBA Valid	0x0	Define the validity of the data value of the PBA in the recovery section: 0b = Data wasn't programmed by firmware. 1b = Data programmed and valid.
1	EETRACK-ID Valid	0x0	Define the validity of the data value of the EE-Track_ID in the recovery section: 0b = Data wasn't programmed by firmware. 1b = Data programmed and valid.
0	PCIR Section Valid	0x0	Define the validity of the PCIR-Data-Recovery section: 0b = Data wasn't programmed by firmware. 1b = Data programmed and valid.

6.2.8.3 Original EE-Track-ID Word 1- 0x0002

The original EE-Track-ID of the device.

Taken from words 0x34-0x35 in the NVM.

If this field is empty (all 0xFF's), firmware must copy the current ETID from words 0x2D-0x2E to words 0x34-0x35 and use this field.

Bits	Field Name	NVM Image Value	Description
15:0	ETID	0x0	

6.2.8.4 Original EE-Track-ID Word 2 0x0003

The original EE-Track-ID of the device.

Taken from words 0x34-0x35 in the NVM.

If this field is empty (all 0xFF's), firmware must copy the current ETID from words 0x2D-0x2E to words 0x34-0x35 and use this field.

Bits	Field Name	NVM Image Value	Description
15:0	ETID	0x0000	



6.2.8.5 PBA[n] - $0x0004 + 1*n, n=0...4$

PBA section.

Bits	Field Name	NVM Image Value	Description
15:0	PBA	0x0	See respective bits in the PRTPM_SAH register.

6.2.8.6 MAC Addresses Length - 0x0009

Length of the MAC addresses field [words].

Bits	Field Name	NVM Image Value	Description
15:0	MAC Addresses Length	0x0010	

6.2.8.7 MAC Addresses Word 0[n] - $0x000A + 3*n, n=0...15$

The PF MAC address table in the NVM that is loaded by firmware to PRTPM_SAL/H. The full table is kept to revert all MAC addresses in SFP and MFP modes.

Bits	Field Name	NVM Image Value	Description
15:0	MAC Addresses Word 0	0x0	

6.2.8.8 MAC Addresses Word 1[n] - $0x000B + 3*n, n=0...15$

The PF MAC address table in the NVM that is loaded by firmware to PRTPM_SAL/H. The full table is kept to revert all MAC addresses in both SFP and MFP modes.

Bits	Field Name	NVM Image Value	Description
15:0	MAC Addresses Word 1	0x0	

6.2.8.9 MAC Addresses Word 2[n] - $(0x000C + 3*n, n=0...15)$

The PF MAC address table in the NVM that is loaded by firmware to PRTPM_SAL/H. The full table is kept to revert all MAC addresses in both SFP and MFP modes.

Bits	Field Name	NVM Image Value	Description
15:0	MAC Addresses Word 2	0x0	



6.2.9 PCIR-Data-Recovery Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	
0x0001 + 1*n, n=0...23	PCIR Word	

6.2.9.1 Section Length - 0x0000

Bits	Field Name	NVM Image Value	Description
15:0	Section Length		Length in: 2 bytes unit - 1 First Section -> Word: PCIR-Data-Recovery Section -> Section Length Last Section -> Word: PCIR-Data-Recovery Section -> PCIR Word

6.2.9.2 PCIR Word[n] - 0x0001 + 1*n, n+0...23

Bits	Field Name	NVM Image Value	Description
15:0	PCIR Field	0x0	

6.2.10 Spare SR Words Section Summary Table

Spare words in SR to separate between RO and RW words.

6.2.11 Permanent SAN MAC Address Section Summary Table

MAC addresses to be used for SAN.

Word Offset	Description	Reference
0x0000	Module Length	Section 6.2.14.1
0x0001 + 3*n, n=0...15	SAN MAC Address Word 0 for PF	Section 6.2.11.2
0x0002 + 3*n, n=0...15	SAN MAC Address Word 1 for PF	Section 6.2.11.3
0x0003 + 3*n, n=0...15	SAN MAC Address Word 2 for PF	Section 6.2.11.4

6.2.11.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		



6.2.11.2 SAN MAC Address Word 0 for PF[n] - $0x0001 + 3*n$, $n=0...15$

PF index n is for PF 0 up to PF 15.

Bits	Field Name	Default NVM Value	Description
15:0	SAN MAC Address Word 0 for PF[n]	0xFFFF	Note: This field is preserved by the Intel NVM update tool.

6.2.11.3 SAN MAC Address Word 1 for PF[n] - $0x0002 + 3*n$, $n=0...15$

PF index n is for PF 0 up to PF 15.

Bits	Field Name	Default NVM Value	Description
15:0	SAN MAC Address Word 1 for PF[n]	0xFFFF	Note: This field is preserved by the Intel NVM update tool.

6.2.11.4 SAN MAC Address Word 2 for PF[n] - $0x0003 + 3*n$, $n=0...15$

PF index n is for PF 0 up to PF 15.

Bits	Field Name	Default NVM Value	Description
15:0	SAN MAC Address Word 2 for PF[n]	0xFFFF	Note: This field is preserved by the Intel NVM update tool.

6.2.12 PBA Block Section Summary Table

The PBA block contains the complete PBA number including the dash and the first digit of the 3-digit suffix.

Word Offset	Description	Reference
0x0000	PBA Section Length	See section 6.2.12.1
0x0001	Word1	See section 6.2.12.2
0x0002	Word2	See section 6.2.12.3
0x0003	Word3	See section 6.2.12.4
0x0004	Word4	See section 6.2.12.5
0x0005	Word5	See section 6.2.12.6



6.2.12.1 PBA Section Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	PBA Section Length Field	0x6	Length in words of the PBA block.

6.2.12.2 Word1 - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Word1 Field		PBA number stored in hexadecimal ASCII values.

6.2.12.3 Word2 - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Word2 Field		PBA number stored in hexadecimal ASCII values.

6.2.12.4 Word3 - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	Word3 Field		PBA number stored in hexadecimal ASCII values.

6.2.12.5 Word4 - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	Word4 Field		PBA number stored in hexadecimal ASCII values.

6.2.12.6 Word5 - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Word5 Field		PBA number stored in hexadecimal ASCII values.

6.2.13 PXE Setup Options Section Summary Table

Setup options for the PXE driver that are defined per PCIe function.

Word Offset	Description	Reference
0x0000	Section Length	Section 6.2.13.1
0x0001 + 1*n, n=0...15	Setup Options PCI Function	Section 6.2.13.2



6.2.13.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		

6.2.13.2 Setup Options PCI Function[n] - 0x0001 + 1*n, n=0...15

The main setup options for PF n are stored in this word. These options are those that can be changed by the user using the Control-S setup menu.

Bits	Field Name	Default NVM Value	Description
15:13	Reserved	0x00	Reserved. Must be 0x0.
12:10	FSD	0x0	Bits 12-10 control forcing speed and duplex during software device driver operation. Valid values are: 000b = Auto-negotiate. 001b = 10 Mb/s half duplex. 010b = 100 Mb/s half duplex. 011b = Not valid (treated as 000b). 100b = 10 Mb/s full duplex. 101b = 100 Mb/s full duplex. 111b = 1000 Mb/s full duplex. Only applicable for copper-based adapters. Not applicable to 10 GbE. Default value is 000b. Valid values are: 0x0 = Auto-negotiate. 0x1 = 10 Mb/s half duplex. 0x2 = 100 Mb/s half duplex. 0x4 = 10 Mb/s full duplex. 0x5 = 100 Mb/s full duplex. 0x7 = 1000 Mb/s full duplex. Note: This field is preserved by the Intel NVM update tool.
9	Reserved	0x0	Reserved to legacy Operating System Wakeup Support. Not supported in the X710-TM4.
8	DSM	0x1	Display Setup Message. If the bit is set to 1b, the Press Control-S message is displayed after the title message. Default value is 1b. Note: This field is preserved by the Intel NVM update tool.
7:6	PT	0x0	Prompt Time. These bits control how long the Press Control-S setup prompt message is displayed, if enabled by DIM. 00b = 2 seconds (default). 01b = 3 seconds. 10b = 5 seconds. 11b = 0 seconds. The Ctrl-S message is not displayed if 0 seconds prompt time is selected. Valid values are: 0x0 = 2 seconds. 0x1 = 3 seconds. 0x2 = 5 seconds. 0x3 = 0 seconds. Note: This field is preserved by the Intel NVM update tool.



Bits	Field Name	Default NVM Value	Description
5	iSCSI Boot Disabled	0x0	<p>When this bit is set and the adapter port is neither iSCSI primary nor secondary, Setup code must not be loaded. Otherwise, iSCSI banner and Setup menu should be accessible as in current design.</p> <p>This bit must be changed at the factory level and not be altered by any end-user tools.</p> <p>Note: For regular NICs and LOM designs this bit should always be cleared in the NVM image; otherwise, iSCSI setup is not accessible for users.</p> <p>0x0 = Enabled. 0x1 = Disabled.</p> <p>Note: This field is preserved by the Intel NVM update tool.</p>
4:3	DBS	0x0	<p>Default Boot Selection. These bits select which device is the default boot device. These bits are only used if boot agent detects that the BIOS does not support boot order selection or if the <i>Mode</i> field of word 0x31 is set to MODE_LEGACY.</p> <p>00b = Network boot, then local boot (default). 01b = Local boot, then network boot. 10b = Network boot only. 11b = Local boot only.</p> <p>Valid values are: 0x0 = Network then local. 0x1 = Local then network. 0x2 = Network only. 0x3 = Local only.</p> <p>Note: This field is preserved by the Intel NVM update tool.</p>
2:0	PS	0x0	<p>Protocol Select. These bits select the active boot protocol.</p> <p>00b = PXE (default value). 01b = RPL (only if RPL is in the Flash). 10b = iSCSI boot primary port (only if iSCSI boot is using this device). 11b = iSCSI boot secondary port (only if iSCSI boot is using this device).</p> <p>Only default value of 00b should be initially programmed into the device; other values should only be set by configuration utilities.</p> <p>Valid values are: 0x0 = PXE (default value). 0x1 = Boot disabled. 0x2 = iSCSI primary. 0x3 = iSCSI secondary. 0x4 = Reserved. 0x5 = Reserved. 0x6 = Reserved. 0x7 = Reserved.</p> <p>Note: This field is preserved by the Intel NVM update tool.</p>



6.2.14 PXE Configuration Customizations Options Section Summary Table

Configuration customization options for the PXE driver that is defined per PCIe function.

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.14.1
0x0001 + 1*n, n=0...15	Configuration Customization Options PCI Function	See section 6.2.14.2

6.2.14.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		

6.2.14.2 Configuration Customization Options PCI Function[n] (0x0001 + 1*n, n=0...15)

Contains settings that can be programmed by an OEM or network administrator to customize the operation of software. These settings cannot be changed from within the Control- S setup menu. The lower byte contains settings that would typically be configured by a network administrator using an external utility; these settings generally control which setup menu options are changeable. The upper byte is generally settings that would be used by an OEM to control the operation of the boot agent in a LOM environment, although there is nothing in the boot agent to prevent their use on a NIC implementation. The default value for this word is 0x4000.

Bits	Field Name	Default NVM Value	Description
15:14	Signature	0x1	Signature. Must be set to 01b to indicate that this word has been programmed by the boot agent or other configuration software. Note: This field is preserved by the Intel NVM update tool.
13:12	Reserved	0x0	Reserved. Must be 0x0.
11	Continuous Retry	0x0	Selects continuous retry operation. If this bit is set, IBA does not transfer control back to the BIOS if it fails to boot due to a network error (such as failure to receive DHCP replies). Instead, it restarts the PXE boot process again. If this bit is set, the only way to cancel PXE boot is for users to press ESC on the keyboard. Retry is not attempted due to hardware conditions such as an invalid NVM checksum or failing to establish link. Default value is 0b. 0x0 = Disable. 0x1 = Enable. Note: This field is preserved by the Intel NVM update tool.



Bits	Field Name	Default NVM Value	Description
10:8	Operating Mode	0x0	<p>Selects the agent’s boot order setup mode. This field changes the boot agent default behavior in order to make it compatible with systems that do not completely support the BBS and PnP expansion ROM standards. Valid values and their meanings are:</p> <p>000b = Normal behavior. The boot agent attempts to detect BBS and PnP expansion ROM support as it normally does.</p> <p>001b = Force legacy mode. The boot agent does not attempt to detect BBS or PnP expansion ROM support in the BIOS and assumes the BIOS is not compliant. Users can change the BIOS boot order in the Setup Menu.</p> <p>010b = Force BBS mode. The agent assumes the BIOS is BBS compliant, even though it might not be detected as such by the agent’s detection code. The user CANNOT change the BIOS boot order in the Setup Menu.</p> <p>011b = Force PnP Int18 mode. The boot agent assumes the BIOS allows boot order setup for PnP expansion ROMs and hooks interrupt 0x18 (to signal the BIOS that the boot agent is a bootable device) in addition to registering as a BBS IPL device. Users cannot change the BIOS boot order in the Setup Menu.</p> <p>100b = Force PnP Int19 mode. The boot agent assumes the BIOS allows boot order setup for PnP expansion ROMs and hooks interrupt 0x19 (to signal the BIOS that the agent is a bootable device) in addition to registering as a BBS IPL device. Users cannot change the BIOS boot order in the Setup Menu.</p> <p>101b = Reserved for future use. If specified, is treated as a value of 000b.</p> <p>110b = Reserved for future use. If specified, is treated as a value of 000b.</p> <p>111b = Reserved for future use. If specified, is treated as a value of 000b.</p> <p>Valid values are: 0x0 = Normal mode. 0x1 = Force legacy. 0x2 = Force BBS. 0x3 = Force PnP Int18. 0x4 = Force PnP Int19. Note: This field is preserved by the Intel NVM update tool.</p>
7:6	Reserved	0x0	Reserved. Must be 0x0.
5	Disable Flash Update	0x0	<p>Disable Flash Update. If this bit is set to 1b, users are not allowed to update the Flash image using PROSet. Default value is 0b.</p> <p>0x0 = Enable Flash update. 0x1 = Disable Flash update. Note: This field is preserved by the Intel NVM update tool.</p>
4	Disable Legacy OS Wakeup Menu	0x0	<p>Disable Legacy OS Wakeup Support. If this bit is set to 1b, users are not allowed to change the Legacy OS Wakeup Support menu option. Default value is 0b.</p> <p>0x0 = Enable legacy wakeup support. 0x1 = Disable legacy wakeup support. Note: This field is preserved by the Intel NVM update tool.</p>
3	Disable Boot Selection Menu	0x0	<p>Disable Boot Selection. If this bit is set to 1b, users are not allowed to change the boot order menu option. Default value is 0b.</p> <p>0x0 = Enable. 0x1 = Disable. Note: This field is preserved by the Intel NVM update tool.</p>
2	Disable Protocol Selection Menu	0x0	<p>Disable Protocol Select. If set to 1b, users are not allowed to change the boot protocol. Default value is 0b.</p> <p>0x0 = Enable. 0x1 = Disable. Note: This field is preserved by the Intel NVM update tool.</p>
1	Disable Title Message Display	0x0	<p>Disable Title Message. If this bit is set to 1b, the title message displaying the version of the boot agent is suppressed. The Control-S message is also suppressed. This is for OEMs who do not want the boot agent to display any messages at system boot. Default value is 0b.</p> <p>0x0 = Enable. 0x1 = Disable. Note: This field is preserved by the Intel NVM update tool.</p>



Bits	Field Name	Default NVM Value	Description
0	Setup Menu	0x0	Disable Setup Menu. If this bit is set to 1b, users are not allowed to invoke the setup menu by pressing Control-S. In this case, the NVM might only be changed via an external program. Default value is 0b. 0x0 = Enable. 0x1 = Disable. Note: This field is preserved by the Intel NVM update tool.

6.2.15 Boot Configuration Block Section Summary Table

Contains the required setup to be used for boot operations.

Word Offset	Description	Reference
0x0000	Boot Signature	See section 6.2.15.1
0x0001	Block Size	See section 6.2.15.2
0x0002	Structure Version	See section 6.2.15.3
0x0003	iSCSI Initiator Name	See section 6.2.15.4
0x0083	Combo Image Version High	See section 6.2.15.5
0x0084	Combo Image Version Low	See section 6.2.15.6
0x0085 + 1*n, n=0...14	Reserved	See section 6.2.15.7
0x0094	iSCSI Flags	See section 6.2.15.8
0x0095 + 1*n, n=0...1	iSCSI Initiator IP	See section 6.2.15.9
0x0097 + 1*n, n=0...1	Subnet Mask	See section 6.2.15.10
0x0099 + 1*n, n=0...1	Gateway IP	See section 6.2.15.11
0x009B	iSCSI Boot LUN	See section 6.2.15.12
0x009C + 1*n, n=0...1	iSCSI Target IP	See section 6.2.15.13
0x009E	iSCSI Target Port	See section 6.2.15.14
0x0168	VLAN ID	See section 6.2.15.15
0x0173 + 1*n, n=0...2	Reserved	See section 6.2.15.16
0x017A	Boot LUN	See section 6.2.15.17
0x017B	VLAN ID	See section 6.2.15.18
0x017C	Target Boot Order	See section 6.2.15.19
0x017D	Reserved	See section 6.2.15.20
0x0182	Boot LUN	See section 6.2.15.21
0x0183	VLAN ID	See section 6.2.15.22
0x0184	Target Boot Order	See section 6.2.15.23
0x0185	Reserved	See section 6.2.15.24
0x018A	Boot LUN	See section 6.2.15.25
0x018B	VLAN ID	See section 6.2.15.26



Word Offset	Description	Reference
0x018C	Target Boot Order	See section 6.2.15.27
0x018D	Reserved	See section 6.2.15.28
0x0192	Boot LUN	See section 6.2.15.29
0x0193	VLAN ID	See section 6.2.15.30
0x0194	Target Boot Order	See section 6.2.15.31
0x0195 + 1*n, n=0...44	Reserved	See section 6.2.15.32
0x01C2	iSCSI Flags	See section 6.2.15.33
0x01C3 + 1*n, n=0...1	iSCSI Initiator IP	See section 6.2.15.34
0x01C5 + 1*n, n=0...1	Subnet Mask	See section 6.2.15.35
0x01C7 + 1*n, n=0...1	Gateway IP	See section 6.2.15.36
0x01C9	iSCSI Boot LUN	See section 6.2.15.37
0x01CA + 1*n, n=0...1	iSCSI Target IP	See section 6.2.15.38
0x01CC	iSCSI Target Port	See section 6.2.15.39
0x0296	VLAN ID	See section 6.2.15.40
0x02A1 + 1*n, n=0...2	Reserved	See section 6.2.15.41
0x02A8	Boot LUN	See section 6.2.15.42
0x02A9	VLAN ID	See section 6.2.15.43
0x02AA	Target Boot Order	See section 6.2.15.44
0x02AB	Reserved	See section 6.2.15.45
0x02B0	Boot LUN	See section 6.2.15.46
0x02B1	VLAN ID	See section 6.2.15.47
0x02B2	Target Boot Order	See section 6.2.15.48
0x02B3	Reserved	See section 6.2.15.49
0x02B8	Boot LUN	See section 6.2.15.50
0x02B9	VLAN ID	See section 6.2.15.51
0x02BA	Target Boot Order	See section 6.2.15.52
0x02BB	Reserved	See section 6.2.15.53
0x02C0	Boot LUN	See section 6.2.15.54
0x02C1	VLAN ID	See section 6.2.15.55
0x02C2	Target Boot Order	See section 6.2.15.56
0x02C3 + 1*n, n=0...44	Reserved	See section 6.2.15.57
0x02F0	iSCSI Flags	See section 6.2.15.58
0x02F1 + 1*n, n=0...1	iSCSI Initiator IP	See section 6.2.15.59
0x02F3 + 1*n, n=0...1	Subnet Mask	See section 6.2.15.60



Word Offset	Description	Reference
0x02F5 + 1*n, n=0...1	Gateway IP	See section 6.2.15.61
0x02F7	iSCSI Boot LUN	See section 6.2.15.62
0x02F8 + 1*n, n=0...1	iSCSI Target IP	See section 6.2.15.63
0x02FA	iSCSI Target Port	See section 6.2.15.64
0x03C4	VLAN ID	See section 6.2.15.65
0x03CF + 1*n, n=0...2	Reserved	See section 6.2.15.66
0x03D6	Boot LUN	See section 6.2.15.67
0x03D7	VLAN ID	See section 6.2.15.68
0x03D8	Target Boot Order	See section 6.2.15.69
0x03D9	Reserved	See section 6.2.15.70
0x03DE	Boot LUN	See section 6.2.15.71
0x03DF	VLAN ID	See section 6.2.15.72
0x03E0	Target Boot Order	See section 6.2.15.73
0x03E1	Reserved	See section 6.2.15.74
0x03E6	Boot LUN	See section 6.2.15.75
0x03E7	VLAN ID	See section 6.2.15.76
0x03E8	Target Boot Order	See section 6.2.15.77
0x03E9	Reserved	See section 6.2.15.78
0x03EE	Boot LUN	See section 6.2.15.79
0x03EF	VLAN ID	See section 6.2.15.80
0x03F0	Target Boot Order	See section 6.2.15.81
0x03F1 + 1*n, n=0...44	Reserved	See section 6.2.15.82
0x041E	iSCSI Flags	See section 6.2.15.83
0x041F + 1*n, n=0...1	iSCSI Initiator IP	See section 6.2.15.84
0x0421 + 1*n, n=0...1	Subnet Mask	See section 6.2.15.85
0x0423 + 1*n, n=0...1	Gateway IP	See section 6.2.15.86
0x0425	iSCSI Boot LUN	See section 6.2.15.87
0x0426 + 1*n, n=0...1	iSCSI Target IP	See section 6.2.15.88
0x0428	iSCSI Target Port	See section 6.2.15.89
0x04F2	VLAN ID	See section 6.2.15.90
0x04FD + 1*n, n=0...2	Reserved	See section 6.2.15.91
0x0504	Boot LUN	See section 6.2.15.92
0x0505	VLAN ID	See section 6.2.15.93
0x0506	Target Boot Order	See section 6.2.15.94



Word Offset	Description	Reference
0x0507	Reserved	See section 6.2.15.95
0x050C	Boot LUN	See section 6.2.15.96
0x050D	VLAN ID	See section 6.2.15.97
0x050E	Target Boot Order	See section 6.2.15.98
0x050F	Reserved	See section 6.2.15.99
0x0514	Boot LUN	See section 6.2.15.100
0x0515	VLAN ID	See section 6.2.15.101
0x0516	Target Boot Order	See section 6.2.15.102
0x0517	Reserved	See section 6.2.15.103
0x051C	Boot LUN	See section 6.2.15.104
0x051D	VLAN ID	See section 6.2.15.105
0x051E	Target Boot Order	See section 6.2.15.106

6.2.15.1 Boot Signature - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Boot Signature	0x5369	Boot Signature: 'i', 'S' (0x5369). Note: This field is preserved by the Intel NVM update tool.

6.2.15.2 Block Size - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Block Size		Length in: 1 byte unit First Section -> Word: Boot Configuration Block -> Boot Signature Last Section -> Word: Boot Configuration Block -> Reserved

6.2.15.3 Structure Version - 0x0002

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x00	Reserved.
7:0	Structure Version	0x01	Version of this structure. Should be set to 1b. Note: This field is preserved by the Intel NVM update tool.

6.2.15.4 iSCSI Initiator Name - 0x0003

Raw data module length: 128 words

iSCSI initiator name.

This field is optional and built by manual input, DHCP host name, or with MAC address.

Note: This word is preserved by Intel NVM update tool.



6.2.15.5 Combo Image Version High - 0x0083

Bits	Field Name	Default NVM Value	Description
15:8	Major	0x0	Combo Image Major Version. Note: This field is preserved by the Intel NVM update tool.
7:0	Build	0x0	Combo Image Build Number High (bits 15:8). Note: This field is preserved by the Intel NVM update tool.

6.2.15.6 Combo Image Version Low - 0x0084

Bits	Field Name	Default NVM Value	Description
15:8	Build	0x0	Combo image Build Number Low (bits 7:0). Note: This field is preserved by the Intel NVM update tool.
7:0	Patch	0x0	Combo Image Patch Level. Note: This field is preserved by the Intel NVM update tool.

6.2.15.7 Reserved[n] - 0x0085 + 1*n, n=0..14

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0000	Reserved for future use, should be set to zero.

6.2.15.8 iSCSI Flags - 0x0094

Bits	Field Name	Default NVM Value	Description
15:10	ARP Timeout	0x0F	Timeout value for each try. Note: This field is preserved by the Intel NVM update tool.
9:8	ARP Retries	0x1	Retry Value. Note: Note: This field is preserved by the Intel NVM update tool.
7:6	Reserved	0x0	Reserved.
5:4	Ctrl-D Entry	0x0	Ctrl-D Entry. 0x0 = Enabled. 0x3 = Disabled Skip Ctrl-D entry. Note: This field is preserved by the Intel NVM update tool.
3:2	Authentication Type	0x0	Authentication Type. 0x0 =None 0x1 = One way CHAP. 0x2 = Mutual CHAP. Note: This field is preserved by the Intel NVM update tool.
1	Enable DHCP for iSCSI Target	0x1	Enable DHCP for getting iSCSI target information. 0x0 = Disabled use static target configuration. 0x1 = Enabled use DHCP to get target information by the option 17 root path. Note: This field is preserved by the Intel NVM update tool.
0	Enable DHCP	0x1	0x0 = Disabled use static configurations from this structure. 0x1 = Enabled overrides configurations retrieved from DHCP. Note: This field is preserved by the Intel NVM update tool.



6.2.15.9 iSCSI Initiator IP[n] - 0x0095 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the initiator IP address.

Set = This field is ignored.

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Initiator IP	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.15.10 Subnet Mask[n] - 0x0097 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the subnet mask.

Set = This field is ignored.

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Initiator IP	0x0000	Note: Note: This field is preserved by the Intel NVM update tool.

6.2.15.11 Gateway IP[n] - 0x0099 + 1*n, n=0...1

Initiator DHCP flag.

not set = This field should contain the gateway IP address.

Set = This field is ignored.

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Initiator IP	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.15.12 iSCSI Boot LUN - 0x009B

Target DHCP flag.

Not set = iSCSI target LUN number should be specified.

Set = This field is ignored.

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Boot LUN	0x0000	Note: This field is preserved by the Intel NVM update tool.



6.2.15.13 iSCSI Target IP[n] - 0x009C + 1*n, n=0...1

Target DHCP flag.

Not set = IP address of iSCSI target.

Set = This field is ignored.

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Initiator IP	0x0000	Note: This field is preserved by the Intel NVM update tool.



6.2.15.14 iSCSI Target Port - 0x009E

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Target Port	0x0CBC	Target DHCP Flag. Not set = TCP port used by iSCSI target. Default is 3260. Set = This field is ignored. Note: This field is preserved by the Intel NVM update tool.

6.2.15.15 VLAN ID - 0x0168

Bits	Field Name	Default NVM Value	Description
15:0	VLAN ID	0x0001	Reserved area since the function is disabled due to Microsoft* restrictions. VLAN ID to include the tag in iSCSI boot frames. A valid VLAN ID is between 1 and 4094. Zero means no VLAN tag support. Note: This field is preserved by the Intel NVM update tool.

6.2.15.16 Reserved[n] - 0x0173 + 1*n, n=0...2

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0000	Reserved for future use, should be set to zero.

6.2.15.17 Boot LUN - 0x017A

Bits	Field Name	Default NVM Value	Description
15:0	Target LUN	0x0000	Target LUN. Note: This field is preserved by the Intel NVM update tool.

6.2.15.18 VLAN ID - 0x017B

Bits	Field Name	Default NVM Value	Description
15:0	VLAN ID	0x0001	VLAN ID for the Port. Default is 0x0. Note: This field is preserved by the Intel NVM update tool.

6.2.15.19 Target Boot Order - 0x017C

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x00	Reserved for future use, should be set to zero.
7:0	Target Boot Order	0x00	Target Boot Order. Valid range is 0-4, with 0 meaning no boot order. Note: This field is preserved by the Intel NVM update tool.

6.2.15.20 Reserved - 0x017D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0000	Reserved.



6.2.15.21 Boot LUN - 0x0182

Bits	Field Name	Default NVM Value	Description
15:0	Target LUN	0x0000	Target LUN. Note: This field is preserved by the Intel NVM update tool.

6.2.15.22 VLAN ID - 0x0183

Bits	Field Name	Default NVM Value	Description
15:0	VLAN ID	0x0001	VLAN ID for the port. Default is zero. Note: This field is preserved by the Intel NVM update tool.

6.2.15.23 Target Boot Order - 0x0184

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x00	Reserved for future use, should be set to zero.
7:0	Target Boot Order	0x00	Target Boot Order. Valid range is 0-4, with 0 meaning no boot order. Note: This field is preserved by the Intel NVM update tool.

6.2.15.24 Reserved - 0x0185

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0000	Reserved.

6.2.15.25 Boot LUN - 0x018A

Bits	Field Name	Default NVM Value	Description
15:0	Target LUN	0x0000	Target LUN. Note: This field is preserved by the Intel NVM update tool.

6.2.15.26 VLAN ID - 0x018B

Bits	Field Name	Default NVM Value	Description
15:0	VLAN ID	0x0001	VLAN ID for the port. Default is zero. Note: This field is preserved by the Intel NVM update tool.

6.2.15.27 Target Boot Order - 0x018C

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x00	Reserved for future use, should be set to zero.
7:0	Target Boot Order	0x00	Target Boot Order. Valid range is 0-4, with 0 meaning no boot order. Note: This field is preserved by the Intel NVM update tool.



6.2.15.28 Reserved - 0x018D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0000	Reserved.

6.2.15.29 Boot LUN - 0x0192

Bits	Field Name	Default NVM Value	Description
15:0	Target LUN	0x0000	Target LUN. Note: This field is preserved by the Intel NVM update tool.

6.2.15.30 VLAN ID - 0x0193

Bits	Field Name	Default NVM Value	Description
15:0	VLAN ID	0x0001	VLAN ID for the port. Default is zero. Note: This field is preserved by the Intel NVM update tool.

6.2.15.31 Target Boot Order - 0x0194

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x00	Reserved for future use, should be set to zero.
7:0	Target Boot Order	0x00	Target Boot Order. Valid range is 0-4, with 0 meaning no boot order. Note: This field is preserved by the Intel NVM update tool.

6.2.15.32 Reserved[n] - 0x0195 + 1*n, n=0..44

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0000	Reserved.

6.2.15.33 iSCSI Flags - 0x01C2

For inner structure detail, see [Section 6.2.15.8](#).

6.2.15.34 iSCSI Initiator IP[n] - 0x01C3 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the initiator IP address.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.9](#).

6.2.15.35 Subnet Mask[n] - 0x01C5 + 1*n, n=0...1

Initiator DHCP flag.



Not set = This field should contain the subnet mask.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.10](#).

6.2.15.36 Gateway IP[n] - 0x01C7 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the gateway IP address.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.11](#).

6.2.15.37 iSCSI Boot LUN - 0x01C9

Target DHCP flag.

Not set = iSCSI target LUN number should be specified.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.12](#).

6.2.15.38 iSCSI Target IP[n] - 0x01CA + 1*n, n=0...1

Target DHCP flag.

Not set = IP address of iSCSI target.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.13](#).

6.2.15.39 iSCSI Target Port - 0x01CC

For inner structure detail, see [Section 6.2.15.14](#).

6.2.15.40 VLAN ID - 0x0296

For inner structure detail, see [Section 6.2.15.15](#).

6.2.15.41 Reserved[n] - 0x02A1 + 1*n, n=0...2

For inner structure detail, see [Section 6.2.15.16](#).

6.2.15.42 Boot LUN - 0x02A8

For inner structure detail, see [Section 6.2.15.17](#).

6.2.15.43 VLAN ID - 0x02A9

For inner structure detail, see [Section 6.2.15.18](#).



6.2.15.44 Target Boot Order - 0x02AA

For inner structure detail, see [Section 6.2.15.19](#).

6.2.15.45 Reserved - 0x02AB

For inner structure detail, see [Section 6.2.15.20](#).

6.2.15.46 Boot LUN - 0x02B0

For inner structure detail, see [Section 6.2.15.21](#).

6.2.15.47 VLAN ID - 0x02B1

For inner structure detail, see [Section 6.2.15.22](#).

6.2.15.48 Target Boot Order - 0x02B2

For inner structure detail, see [Section 6.2.15.23](#).

6.2.15.49 Reserved - 0x02B3

For inner structure detail, see [Section 6.2.15.24](#).

6.2.15.50 Boot LUN - 0x02B8

For inner structure detail, see [Section 6.2.15.25](#).

6.2.15.51 VLAN ID - 0x02B9

For inner structure detail, see [Section 6.2.15.26](#).

6.2.15.52 Target Boot Order - 0x02BA

For inner structure detail, see [Section 6.2.15.27](#).

6.2.15.53 Reserved - 0x02BB

For inner structure detail, see [Section 6.2.15.28](#).

6.2.15.54 Boot LUN - 0x02C0

For inner structure detail, see [Section 6.2.15.29](#).

6.2.15.55 VLAN ID - 0x02C1

For inner structure detail, see [Section 6.2.15.30](#).

6.2.15.56 Target Boot Order - 0x02C2

For inner structure detail, see [Section 6.2.15.31](#).



6.2.15.57 Reserved[n] - 0x02C3 + 1*n, n=0...44

For inner structure detail, see [Section 6.2.15.32](#).

6.2.15.58 iSCSI Flags - 0x02F0

For inner structure detail, see [Section 6.2.15.8](#).

6.2.15.59 iSCSI Initiator IP[n] - 0x02F1 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the initiator IP address.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.9](#).



6.2.15.60 Subnet Mask[n] - 0x02F3 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the subnet mask.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.10](#).

6.2.15.61 Gateway IP[n] - 0x02F5 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the gateway IP address.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.11](#).

6.2.15.62 iSCSI Boot LUN - 0x02F7

Target DHCP flag.

Not set = iSCSI target LUN number should be specified.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.12](#).

6.2.15.63 iSCSI Target IP[n] - 0x02F8 + 1*n, n=0...1

Target DHCP flag.

Not set = IP address of iSCSI target.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.13](#).

6.2.15.64 iSCSI Target Port - 0x02FA

For inner structure detail, see [Section 6.2.15.14](#).

6.2.15.65 VLAN ID - 0x03C4

For inner structure detail, see [Section 6.2.15.15](#).

6.2.15.66 Reserved[n] - 0x03CF + 1*n, n=0...2

For inner structure detail, see [Section 6.2.15.16](#).

6.2.15.67 Boot LUN - 0x03D6

For inner structure detail, see [Section 6.2.15.17](#).



6.2.15.68 VLAN ID - 0x03D7

For inner structure detail, see [Section 6.2.15.18](#).

6.2.15.69 Target Boot Order - 0x03D8

For inner structure detail, see [Section 6.2.15.19](#).

6.2.15.70 Reserved - 0x03D9

For inner structure detail, see [Section 6.2.15.20](#).

6.2.15.71 Boot LUN - 0x03DE

For inner structure detail, see [Section 6.2.15.21](#).

6.2.15.72 VLAN ID - 0x03DF

For inner structure detail, see [Section 6.2.15.22](#).

6.2.15.73 Target Boot Order - 0x03E0

For inner structure detail, see [Section 6.2.15.23](#).

6.2.15.74 Reserved - 0x03E1

For inner structure detail, see [Section 6.2.15.24](#).

6.2.15.75 Boot LUN - 0x03E6

For inner structure please See [section 6.2.15.25](#)

6.2.15.76 VLAN ID - 0x03E7

For inner structure please See [section 6.2.15.26](#)

6.2.15.77 Target Boot Order - 0x03E8

For inner structure please See [section 6.2.15.27](#)

6.2.15.78 Reserved - 0x03E9

For inner structure please See [section 6.2.15.28](#)

6.2.15.79 Boot LUN - 0x03EE

For inner structure detail, see [Section 6.2.15.29](#).

6.2.15.80 VLAN ID - 0x03EF

For inner structure detail, see [Section 6.2.15.30](#).



6.2.15.81 Target Boot Order - 0x03F0

For inner structure detail, see [Section 6.2.15.31](#).

6.2.15.82 Reserved[n] - 0x03F1 + 1*n, n=0...44

For inner structure detail, see [Section 6.2.15.32](#).

6.2.15.83 iSCSI Flags - 0x041E

For inner structure detail, see [Section 6.2.15.8](#).



6.2.15.84 iSCSI Initiator IP[n] - 0x041F + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the initiator IP address.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.9](#).

6.2.15.85 Subnet Mask[n] - 0x0421 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the subnet mask.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.10](#).

6.2.15.86 Gateway IP[n] - 0x0423 + 1*n, n=0...1

Initiator DHCP flag.

Not set = This field should contain the gateway IP address.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.11](#).

6.2.15.87 iSCSI Boot LUN - 0x0425

Target DHCP flag.

Not set = iSCSI target LUN number should be specified.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.12](#).

6.2.15.88 iSCSI Target IP[n] - 0x0426 + 1*n, n=0...1

Target DHCP flag.

Not set = IP address of iSCSI target.

Set = This field is ignored.

For inner structure detail, see [Section 6.2.15.13](#).

6.2.15.89 iSCSI Target Port - 0x0428

For inner structure detail, see [Section 6.2.15.14](#).



6.2.15.90 VLAN ID - 0x04F2

For inner structure detail, see [Section 6.2.15.15](#).

6.2.15.91 Reserved[n] - 0x04FD + 1*n, n=0...2

For inner structure detail, see [Section 6.2.15.16](#).

6.2.15.92 Boot LUN - 0x0504

For inner structure detail, see [Section 6.2.15.17](#).

6.2.15.93 VLAN ID - 0x0505

For inner structure detail, see [Section 6.2.15.18](#).

6.2.15.94 Target Boot Order - 0x0506

For inner structure detail, see [Section 6.2.15.19](#).

6.2.15.95 Reserved - 0x0507

For inner structure detail, see [Section 6.2.15.20](#).

6.2.15.96 Boot LUN - 0x050C

For inner structure detail, see [Section 6.2.15.21](#).

6.2.15.97 VLAN ID - 0x050D

For inner structure detail, see [Section 6.2.15.22](#).

6.2.15.98 Target Boot Order - 0x050E

For inner structure detail, see [Section 6.2.15.23](#).

6.2.15.99 Reserved - 0x050F

For inner structure detail, see [Section 6.2.15.24](#).

6.2.15.100 Boot LUN - 0x0514

For inner structure detail, see [Section 6.2.15.25](#).

6.2.15.101 VLAN ID - 0x0515

For inner structure detail, see [Section 6.2.15.26](#).

6.2.15.102 Target Boot Order - 0x0516

For inner structure detail, see [Section 6.2.15.27](#).



6.2.15.103 Reserved - 0x0517

For inner structure detail, see [Section 6.2.15.28](#).

6.2.15.104 Boot LUN - 0x051C

For inner structure detail, see [Section 6.2.15.29](#).

6.2.15.105 VLAN ID - 0x051D

For inner structure detail, see [Section 6.2.15.30](#).

6.2.15.106 Target Boot Order - 0x051E

For inner structure detail, see [Section 6.2.15.31](#).

6.2.15.107 Reserved[n] - 0x051F + 1*n, n=0...44

For inner structure detail, see [Section 6.2.15.32](#).

6.2.16 VLAN Configuration Block Section Summary Table

This section contains the allocation of resources per PF which shall be stored in the shadow RAM.

Word Offset	Description	Reference
0x0000	Section Header	See section 6.2.16.1
0x0001	VLAN Block Signature	See section 6.2.16.2
0x0002	Structure Version and Size	See section 6.2.16.3
0x0003	Port 0 VLAN Tag	See section 6.2.16.4
0x0004	Port 1 VLAN Tag	See section 6.2.16.5
0x0005	Port 2 VLAN Tag	See section 6.2.16.6
0x0006	Port 3 VLAN Tag	See section 6.2.16.7



6.2.16.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Block Length		Length in: 2 Bytes unit - 1 First Section -> Word: VLAN Configuration Block -> Section Header Last Section -> Word: VLAN Configuration Block -> Port 3 VLAN Tag Section Length in Words

6.2.16.2 VLAN Block Signature - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	VLAN block signature	0x4C56	'V' = 0x56. 'L' = 0x4C.

6.2.16.3 Structure Version and Size - 0x0002

Bits	Field Name	Default NVM Value	Description
15:8	Structure Version and Size	0x0C	Total byte size of the configuration block. This is 0x6 for 1-port adapter, 0x8 for 2-port adapter and 0xC for 4-port adapter (the default value).
7:0	Structure Version	0x01	The version of this structure. Should be set to 0x1.

6.2.16.4 Port 0 VLAN Tag - 0x0003

Bits	Field Name	Default NVM Value	Description
15:13	VLAN Priority	0x0	The value of VLAN priority (0-7).
12	Reserved	0x0	This field is reserved and must be set to 0x0.
11:0	VLAN Tag ID	0x0	The value of VLAN ID (1-4095). Zero means no VLAN configured for port.

6.2.16.5 Port 1 VLAN Tag - 0x0004

For inner structure detail, see [Section 6.2.16.4](#).

6.2.16.6 Port 2 VLAN Tag - 0x0005

For inner structure detail, see [Section 6.2.16.4](#).

6.2.16.7 Port 3 VLAN Tag - 0x0006

For inner structure detail, see [Section 6.2.16.4](#).



6.2.17 PCIR Registers Auto-load Module Section Summary Table

Default setup to registers and internal memories that load on PCIR events.

Word Offset	Description	Reference
0x0000	Module Length	See section 6.2.17.1
0x0001 - 0023	NVM contents for PFPCI_CNF	See section 6.2.17.2
0x0024 - 0046	NVM contents for PFPCI_DEVID	See section 6.2.17.3
0x0047 - 0069	NVM contents for PFPCI_SUBSYSID	See section 6.2.17.4
0x006A - 008C	NVM contents for PFPCI_FUNC2	See section 6.2.17.5
0x008D - 00AF	NVM contents for PF_VT_PFALLOC_PCIE	See section 6.2.17.6
0x00B0 - 00D2	NVM contents for PFPCI_CLASS	See section 6.2.17.7
0x00D3 - 00D6	NVM contents for GLTPH_CTRL	See section 6.2.17.8
0x00D7 - 00DB	NVM contents for GLPCI_SUBVENID	See section 6.2.17.9
0x00DC - 00DD	NVM contents for GLPCI_PWRDATA	See section 6.2.17.10
0x00DE - 00DF	NVM contents for GLPCI_CNF2	See section 6.2.17.11
0x00E0 - 00E1	NVM contents for GLPCI_SERL	See section 6.2.17.12
0x00E2 - 00E3	NVM contents for GLPCI_SERH	See section 6.2.17.13
0x00E4 - 00E8	NVM contents for GLPCI_CAPCTRL	See section 6.2.17.14
0x00E9 - 00EA	NVM contents for GLPCI_CAPSUP	See section 6.2.17.15
0x00EB - 00EC	NVM contents for GLPCI_LINKCAP	See section 6.2.17.16
0x00ED - 00EE	NVM contents for GLPCI_PMSUP	See section 6.2.17.17
0x00EF - 00F0	NVM contents for GLPCI_REVID	See section 6.2.17.18
0x00F1 - 00F2	NVM contents for GLPCI_VFSUP	See section 6.2.17.19
0x00FB - 00FE	NVM contents for GLPCI_VENDORID	See section 6.2.17.20
0x00FF - 0102	NVM contents for GLGEN_PCIFCNCNT	See section 6.2.17.21

6.2.17.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		Length in: 2 bytes unit - 1 First Section -> Word: PCIR Registers Auto-load Module -> Module Length Last Section -> Word: PCIR Registers Auto-load Module -> Address Low at GLGEN_PCIFCNCNT

6.2.17.2 PFPCI_CNF - (0x0001 - 0023)

6.2.17.2.1 Starting Address Low at PFPCI_CNF - 0x0001

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFPCI_CNF for PF[0]	0xBE000	
3:0	Type	0x2	



6.2.17.2.2 Starting Address High at PFPCI_CNF - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFPCI_CNF for PF[0]		

6.2.17.2.3 Attributes at PFPCI_CNF - 0x0003

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.17.2.4 Data Low of PFPCI_CNF[PF] - 0x0004 + 2*PF, PF=0...15

6.2.17.2.5 Data High of PFPCI_CNF[PF] - 0x0005 + 2*PF, PF=0...15

6.2.17.3 PFPCI_DEVID - (0x0024 - 0046)

6.2.17.3.1 Starting Address Low at PFPCI_DEVID - 0x0024

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFPCI_DEVID for PF[0]	0xBE080	
3:0	Type	0x2	

6.2.17.3.2 Starting Address High at PFPCI_DEVID - 0x0025

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFPCI_DEVID for PF[0]		

6.2.17.3.3 Attributes at PFPCI_DEVID - 0x0026

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.17.3.4 Data Low of PFPCI_DEVID[PF] - 0x0027 + 2*PF, PF=0...15

6.2.17.3.5 Data High of PFPCI_DEVID[PF] - 0x0028 + 2*PF, PF=0...15



6.2.17.4 PFPCI_SUBSYSID - (0x0047 - 0069)

6.2.17.4.1 Starting Address Low at PFPCI_SUBSYSID - 0x0047

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFPCI_SUBSYSID for PF[0]	0xBE100	
3:0	Type	0x2	

6.2.17.4.2 Starting Address High at PFPCI_SUBSYSID - 0x0048

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFPCI_SUBSYSID for PF[0]		

6.2.17.4.3 Attributes at PFPCI_SUBSYSID - 0x0049

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.17.4.4 Data Low of PFPCI_SUBSYSID[PF] - $0x004A + 2*PF$, PF=0...15

6.2.17.4.5 Data High of PFPCI_SUBSYSID[PF] - $0x004B + 2*PF$, PF=0...15

6.2.17.5 PFPCI_FUNC2 - (0x006A - 008C)

6.2.17.5.1 Starting Address Low at PFPCI_FUNC2 - 0x006A

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFPCI_FUNC2 for PF[0]	0xBE180	
3:0	Type	0x2	

6.2.17.5.2 Starting Address High at PFPCI_FUNC2 - 0x006B

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFPCI_FUNC2 for PF[0]		



6.2.17.5.3 Attributes at PFPCI_FUNC2 - 0x006C

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.17.5.4 Data Low of PFPCI_FUNC2[PF] - 0x006D + 2*PF, PF=0...15

6.2.17.5.5 Data High of PFPCI_FUNC2[PF] - 0x006E + 2*PF, PF=0...15

6.2.17.6 PF_VT_PFALLOC_PCIE - (0x008D - 00AF)

6.2.17.6.1 Starting Address Low at PF_VT_PFALLOC_PCIE - 0x008D

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PF_VT_PFALLOC_PCIE for PF[0]	0xBE380	
3:0	Type	0x2	

6.2.17.6.2 Starting Address High at PF_VT_PFALLOC_PCIE - 0x008E

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PF_VT_PFALLOC_PCIE for PF[0]		

6.2.17.6.3 Attributes at PF_VT_PFALLOC_PCIE - 0x008F

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.17.7 PFPCI_CLASS - (0x00B0 - 00D2)

6.2.17.7.1 Starting Address Low at PFPCI_CLASS - 0x00B0

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFPCI_CLASS for PF[0]	0xBE400	
3:0	Type	0x2	



6.2.17.7.2 Starting Address High at PFPCI_CLASS - 0x00B1

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFPCI_CLASS for PF[0]		

6.2.17.7.3 Attributes at PFPCI_CLASS - 0x00B2

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.17.7.4 Data Low of PFPCI_CLASS[PF] - 0x00B3 + 2*PF, PF=0...15

6.2.17.7.5 Data High of PFPCI_CLASS[PF] - 0x00B4 + 2*PF, PF=0...15



6.2.17.8 GLTPH_CTRL - (0x00D3 - 00D6)

6.2.17.8.1 Address Low at GLTPH_CTRL - 0x00D3

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLTPH_CTRL	0xBE480	
3:0	Type	0x1	

6.2.17.8.2 Address High at GLTPH_CTRL - 0x00D4

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLTPH_CTRL		

6.2.17.8.3 Data Low of GLTPH_CTRL - 0x00D5

6.2.17.8.4 Data High of GLTPH_CTRL - 0x00D6

6.2.17.9 GLPCI_SUBVENID - (0x00D7 - 00DB)

6.2.17.9.1 Starting Address Low at GLPCI_SUBVENID - 0x00D7

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPCI_SUBVENID	0xBE48C	
3:0	Type	0x2	

6.2.17.9.2 Starting Address High at GLPCI_SUBVENID - 0x00D8

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPCI_SUBVENID		

6.2.17.9.3 Attributes at GLPCI_SUBVENID - 0x00D9

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x5	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.17.9.4 Data Low of GLPCI_SUBVENID - 0x00DA

6.2.17.9.5 Data High of GLPCI_SUBVENID - 0x00DB



6.2.17.10 GLPCI_PWRDATA - (0x00DC - 00DD)

6.2.17.10.1 Data Low of GLPCI_PWRDATA - 0x00DC

6.2.17.10.2 Data High of GLPCI_PWRDATA - 0x00DD

6.2.17.11 GLPCI_CNF2 - (0x00DE - 00DF)

6.2.17.11.1 Data Low of GLPCI_CNF2 - 0x00DE

6.2.17.11.2 Data High of GLPCI_CNF2 - 0x00DF

6.2.17.12 GLPCI_SERL - (0x00E0 - 00E1)

6.2.17.12.1 Data Low of GLPCI_SERL - 0x00E0

6.2.17.12.2 Data High of GLPCI_SERL - 0x00E1

6.2.17.13 GLPCI_SERH - (0x00E2 - 00E3)

6.2.17.13.1 Data Low of GLPCI_SERH - 0x00E2

6.2.17.13.2 Data High of GLPCI_SERH - 0x00E3

6.2.17.14 GLPCI_CAPCTRL - (0x00E4 - 00E8)

6.2.17.14.1 Starting Address Low at GLPCI_CAPCTRL - 0x00E4

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPCI_CAPCTRL	0xBE4A4	
3:0	Type	0x2	

6.2.17.14.2 Starting Address High at GLPCI_CAPCTRL - 0x00E5

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPCI_CAPCTRL		

6.2.17.14.3 Attributes at GLPCI_CAPCTRL - 0x00E6

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x6	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.17.14.4 Data Low of GLPCI_CAPCTRL - 0x00E7

6.2.17.14.5 Data High of GLPCI_CAPCTRL - 0x00E8

6.2.17.15 GLPCI_CAPSUP - (0x00E9 - 00EA)

6.2.17.15.1 Data Low of GLPCI_CAPSUP - 0x00E9

6.2.17.15.2 Data High of GLPCI_CAPSUP - 0x00EA

6.2.17.16 GLPCI_LINKCAP - (0x00EB - 00EC)

6.2.17.16.1 Data Low of GLPCI_LINKCAP - 0x00EB

6.2.17.16.2 Data High of GLPCI_LINKCAP - 0x00EC

6.2.17.17 GLPCI_PMSUP - (0x00ED - 00EE)

6.2.17.17.1 Data Low of GLPCI_PMSUP - 0x00ED

6.2.17.17.2 Data High of GLPCI_PMSUP - 0x00EE

6.2.17.18 GLPCI_REVID - (0x00EF - 00F0)

6.2.17.18.1 Data Low of GLPCI_REVID - 0x00EF

6.2.17.18.2 Data High of GLPCI_REVID - 0x00F0

6.2.17.19 GLPCI_VFSUP - (0x00F1 - 00F2)

6.2.17.19.1 Data Low of GLPCI_VFSUP - 0x00F1

6.2.17.19.2 Data High of GLPCI_VFSUP - 0x00F2

6.2.17.20 GLPCI_VENDORID - (0x00FB - 00FE)

6.2.17.20.1 Address Low at GLPCI_VENDORID - 0x00FB

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPCI_VENDORID	0xBE518	
3:0	Type	0x1	



6.2.17.20.2 Address High at GLPCI_VENDORID - 0x00FC

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPCI_VENDORID		



6.2.17.20.3 Data Low of GLPCI_VENDORID - 0x00FD

6.2.17.20.4 Data High of GLPCI_VENDORID - 0x00FE

6.2.17.21 GLGEN_PCIFCNCNT - (0x00FF - 0102)

6.2.17.21.1 Address Low at GLGEN_PCIFCNCNT - 0x00FF

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLGEN_PCIFCNCNT	0x1C0AB4	
3:0	Type	0x1	

6.2.17.21.2 Address High at GLGEN_PCIFCNCNT - 0x0100

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLGEN_PCIFCNCNT		

6.2.17.21.3 Data Low of GLGEN_PCIFCNCNT - 0x0101

6.2.17.21.4 Data High of GLGEN_PCIFCNCNT - 0x0102

6.2.18 PCIR Registers Auto-load Module Section Summary Table

Word Offset	Description	Reference
0x0002	LCB Attributes	See section 6.2.18.1
0x0003	LCB Data	See section 6.2.18.2

6.2.18.1 LCB Attributes - 0x0002

Part of a Type 4 structure to load the PCIe LCB unit.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x060	
4:3	Skip	0x00	

6.2.18.2 LCB Data - 0x0003

Raw data module length: 256 words.

Part of a Type 4 structure to load the PCIe LCB unit.



6.2.19 POR Registers Auto-load Module Section Summary Table

Default setup to registers that load on POR events.

Word Offset	Description	Reference
0x0000	Module Length	See section 6.2.19.1
0x0001	Starting Address Low of PRTPM_GC	See section 6.2.19.2
0x0002	Starting Address High of PRTPM_GC	See section 6.2.19.3
0x0003	Attributes of PRTPM_GC	See section 6.2.19.4
0x0004	Data Low of PRTPM_GC[0]	See section 6.2.19.5
0x0005	Data High of PRTPM_GC[0]	See section 6.2.19.6
0x0006	Data Low of PRTPM_GC[1]	See section 6.2.19.7
0x0007	Data High of PRTPM_GC[1]	See section 6.2.19.8
0x0008	Data Low of PRTPM_GC[2]	See section 6.2.19.9
0x0009	Data High of PRTPM_GC[2]	See section 6.2.19.10
0x000A	Data Low of PRTPM_GC[3]	See section 6.2.19.11
0x000B	Data High of PRTPM_GC[3]	See section 6.2.19.12
0x000C - 002E	NVM contents for PFPM_WUC	See section 6.2.19.13
0x002F - 0032	NVM contents for GLPM_WUMC	See section 6.2.19.14
0x0033 - 0071	NVM contents for GLGEN_GPIO_CTL	See section 6.2.19.15
0x0072 - 0073	NVM contents for GLGEN_LED_CTL	See section 6.2.19.16
0x0074 - 007E	NVM contents for GLGEN_I2CPARAMS	See section 6.2.19.17
0x007F - 0089	NVM contents for GLGEN_MDIO_I2C_SEL	See section 6.2.19.18
0x008A - 0091	NVM contents for GLGEN_MDIO_CTRL	See section 6.2.19.19
0x0092 - 0095	NVM contents for GLNVM_FLASHID	See section 6.2.19.20
0x0096 - 0099	NVM contents for GL_MNG_HWARB_CTRL	See section 6.2.19.21
0x009E - 00C0	NVM contents for PFGEN_PORTNUM_CAR	See section 6.2.19.22
0x00C1 - 00E3	NVM contents for PFPM_APM	See section 6.2.19.23
0x00E4 - 00EE	NVM contents for PRTGEN_CNF	See section 6.2.19.24
0x00EF - 00F9	NVM contents for PRTPM_GC	See section 6.2.19.25
0x00FA - 0104	NVM contents for PRTGEN_CNF2	See section 6.2.19.26
0x010D - 012F	NVM contents for PFPCI_FUNC	See section 6.2.19.27
0x0130 - 0133	NVM contents for GLPCI_CNF	See section 6.2.19.28



6.2.19.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		Length in: 2 bytes unit - 1 First Section -> Word: POR Registers Auto-load Module -> Module Length Last Section -> Word: POR Registers Auto-load Module -> Address Low at GLPCI_SPARE1

6.2.19.2 Starting Address Low of PRTPM_GC - 0x0001

Bits	Field Name	Default NVM Value	Description
15:4	Starting Address Low of PRTPM_GC	0x140	
3:0	Type	0x2	

6.2.19.3 Starting Address High of PRTPM_GC - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Starting Address High of PRTPM_GC	0x00B8	

6.2.19.4 Attributes of PRTPM_GC - 0x0003

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.5 Data Low of PRTPM_GC[0] - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	Data	0x0003	

6.2.19.6 Data High of PRTPM_GC[0] - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Data	0x0000	

6.2.19.7 Data Low of PRTPM_GC[1] - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	Data	0x0003	



6.2.19.8 Data High of PRTPM_GC[1] - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	Data	0x0000	

6.2.19.9 Data Low of PRTPM_GC[2] - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	Data	0x0003	

6.2.19.10 Data High of PRTPM_GC[2] - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	Data	0x0000	

6.2.19.11 Data Low of PRTPM_GC[3] - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	Data	0x0003	

6.2.19.12 Data High of PRTPM_GC[3] - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	Data	0x0000	

6.2.19.13 PFPM_WUC - (0x000C - 002E)

6.2.19.13.1 Starting Address Low at PFPM_WUC - 0x000C

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFPM_WUC for PF[0]	0x6B200	
3:0	Type	0x2	

6.2.19.13.2 Starting Address High at PFPM_WUC - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFPM_WUC for PF[0]		



6.2.19.13.3 Attributes at PFPW_WUC - 0x000E

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.13.4 Data Low of PFPW_WUC[PF] - 0x000F + 2*PF, PF=0...15

6.2.19.13.5 Data High of PFPW_WUC[PF] - 0x0010 + 2*PF, PF=0...15

6.2.19.14 GLPM_WUMC - (0x002F - 0032)

6.2.19.14.1 Address Low at GLPM_WUMC - 0x002F

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPM_WUMC	0x6C800	
3:0	Type	0x1	

6.2.19.14.2 Address High at GLPM_WUMC - 0x0030

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPM_WUMC		

6.2.19.14.3 Data Low of GLPM_WUMC - 0x0031

6.2.19.14.4 Data High of GLPM_WUMC - 0x0032

6.2.19.15 GLGEN_GPIO_CTL - (0x0033 - 0071)

6.2.19.15.1 Starting Address Low at GLGEN_GPIO_CTL - 0x0033

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLGEN_GPIO_CTL	0x88100	
3:0	Type	0x2	

6.2.19.15.2 Starting Address High at GLGEN_GPIO_CTL - 0x0034

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLGEN_GPIO_CTL		



6.2.19.15.3 Attributes at GLGEN_GPIO_CTL - 0x0035

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x1F	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.15.4 Data Low of GLGEN_GPIO_CTL[n] - 0x0036 + 2*n, n=0...29

6.2.19.15.5 Data High of GLGEN_GPIO_CTL[n] - 0x0037 + 2*n, n=0...29

6.2.19.16 GLGEN_LED_CTL - (0x0072 - 0073)

6.2.19.16.1 Data Low of GLGEN_LED_CTL - 0x0072

6.2.19.16.2 Data High of GLGEN_LED_CTL - 0x0073

6.2.19.17 GLGEN_I2CPARAMS - (0x0074 - 007E)

6.2.19.17.1 Starting Address Low at GLGEN_I2CPARAMS - 0x0074

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLGEN_I2CPARAMS	0x881AC	
3:0	Type	0x2	

6.2.19.17.2 Starting Address High at GLGEN_I2CPARAMS - 0x0075

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLGEN_I2CPARAMS		

6.2.19.17.3 Attributes at GLGEN_I2CPARAMS - 0x0076

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.17.4 Data Low of GLGEN_I2CPARAMS[n] - 0x0077 + 2*n, n=0...3

6.2.19.17.5 Data High of GLGEN_I2CPARAMS[n] - 0x0078 + 2*n, n=0...3



6.2.19.18 GLGEN_MDIO_I2C_SEL - (0x007F - 0089)

6.2.19.18.1 Starting Address Low at GLGEN_MDIO_I2C_SEL - 0x007F

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLGEN_MDIO_I2C_SEL	0x881C0	
3:0	Type	0x2	

6.2.19.18.2 Starting Address High at GLGEN_MDIO_I2C_SEL - 0x0080

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLGEN_MDIO_I2C_SEL		

6.2.19.18.3 Attributes at GLGEN_MDIO_I2C_SEL - 0x0081

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x8	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.18.4 Data Low of GLGEN_MDIO_I2C_SEL[n] - $0x0082 + 2*n$, $n=0...3$

6.2.19.18.5 Data High of GLGEN_MDIO_I2C_SEL[n] - $0x0083 + 2*n$, $n=0...3$

6.2.19.19 GLGEN_MDIO_CTRL - (0x008A - 0091)

6.2.19.19.1 Data Low of GLGEN_MDIO_CTRL[n] - $0x008A + 2*n$, $n=0...3$

6.2.19.19.2 Data High of GLGEN_MDIO_CTRL[n] - $0x008B + 2*n$, $n=0...3$

6.2.19.20 GLNVM_FLASHID - (0x0092 - 0095)

6.2.19.20.1 Address Low at GLNVM_FLASHID - 0x0092

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLNVM_FLASHID	0xB6104	
3:0	Type	0x1	



6.2.19.20.2 Address High at GLNVM_FLASHID - 0x0093

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLNVM_FLASHID		

6.2.19.20.3 Data Low of GLNVM_FLASHID - 0x0094

6.2.19.20.4 Data High of GLNVM_FLASHID - 0x0095

6.2.19.21 GL_MNG_HWARB_CTRL - (0x0096 - 0099)

6.2.19.21.1 Address Low at GL_MNG_HWARB_CTRL - 0x0096

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GL_MNG_HWARB_CTRL	0xB6130	
3:0	Type	0x1	

6.2.19.21.2 Address High at GL_MNG_HWARB_CTRL - 0x0097

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GL_MNG_HWARB_CTRL		

6.2.19.21.3 Data Low of GL_MNG_HWARB_CTRL - 0x0098

6.2.19.21.4 Data High of GL_MNG_HWARB_CTRL - 0x0099

6.2.19.22 PFGEN_PORTNUM_CAR - (0x009E - 00C0)

6.2.19.22.1 Starting Address Low at PFGEN_PORTNUM_CAR - 0x009E

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFGEN_PORTNUM_CAR for PF[0]	0xB8000	
3:0	Type	0x2	

6.2.19.22.2 Starting Address High at PFGEN_PORTNUM_CAR - 0x009F

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFGEN_PORTNUM_CAR for PF[0]		



6.2.19.22.3 Attributes at PFGEN_PORTNUM_CAR - 0x00A0

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.23 PFPM_APM - (0x00C1 - 00E3)

6.2.19.23.1 Starting Address Low at PFPM_APM - 0x00C1

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFPM_APM for PF[0]	0xB8080	
3:0	Type	0x2	

6.2.19.23.2 Starting Address High at PFPM_APM - 0x00C2

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFPM_APM for PF[0]		

6.2.19.23.3 Attributes at PFPM_APM - 0x00C3

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.23.4 Data Low of PFPM_APM[PF] - 0x00C4 + 2*PF, PF=0...15

6.2.19.23.5 Data High of PFPM_APM[PF] - 0x00C5 + 2*PF, PF=0...15

6.2.19.24 PRTGEN_CNF - (0x00E4 - 00EE)

6.2.19.24.1 Starting Address Low at PRTGEN_CNF - 0x00E4

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTGEN_CNF for PRT[0]	0xB8120	
3:0	Type	0x2	



6.2.19.24.2 Starting Address High at PRTGEN_CNF - 0x00E5

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTGEN_CNF for PRT[0]		

6.2.19.24.3 Attributes at PRTGEN_CNF - 0x00E6

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.24.4 Data Low of PRTGEN_CNF[PRT] - 0x00E7 + 2*PRT, PRT=0...3

6.2.19.24.5 Data High of PRTGEN_CNF[PRT] - 0x00E8 + 2*PRT, PRT=0...3

6.2.19.25 PRTPM_GC - (0x00EF - 00F9)

6.2.19.25.1 Starting Address Low at PRTPM_GC - 0x00EF

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTPM_GC for PRT[0]	0xB8140	
3:0	Type	0x2	

6.2.19.25.2 Starting Address High at PRTPM_GC - 0x00F0

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTPM_GC for PRT[0]		

6.2.19.25.3 Attributes at PRTPM_GC - 0x00F1

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.25.4 Data Low of PRTPM_GC[PRT] - 0x00F2 + 2*PRT, PRT=0...3

6.2.19.25.5 Data High of PRTPM_GC[PRT] - 0x00F3 + 2*PRT, PRT=0...3



6.2.19.26 PRTGEN_CNF2 - (0x00FA - 0104)

6.2.19.26.1 Starting Address Low at PRTGEN_CNF2] - 0x00FA

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTGEN_CNF2 for PRT[0]	0xB8160	
3:0	Type	0x2	

6.2.19.26.2 Starting Address High at PRTGEN_CNF2 - 0x00FB

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTGEN_CNF2 for PRT[0]		

6.2.19.26.3 Attributes at PRTGEN_CNF2 - 0x00FC

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.26.4 Data Low of PRTGEN_CNF2[PRT] - 0x00FD + 2*PRT, PRT=0...3

6.2.19.26.5 Data High of PRTGEN_CNF2[PRT] - 0x00FE + 2*PRT, PRT=0...3

6.2.19.27 PFPCI_FUNC - (0x010D - 012F)

6.2.19.27.1 Starting Address Low at PFPCI_FUNC - 0x010D

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFPCI_FUNC for PF[0]	0xBE200	
3:0	Type	0x2	

6.2.19.27.2 Starting Address High at PFPCI_FUNC - 0x010E

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFPCI_FUNC for PF[0]		



6.2.19.27.3 Attributes at PFPCI_FUNC - 0x010F

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.19.27.4 Data Low of PFPCI_FUNC[PF] - 0x0110 + 2*PF, PF=0...15

6.2.19.27.5 Data High of PFPCI_FUNC[PF] - 0x0111 + 2*PF, PF=0...15

6.2.19.28 GLPCI_CNF - (0x0130 - 0133)

6.2.19.28.1 Address Low at GLPCI_CNF - 0x0130

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPCI_CNF	0xBE4C0	
3:0	Type	0x1	

6.2.19.28.2 Address High at GLPCI_CNF - 0x0131

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPCI_CNF		

6.2.19.28.3 Data Low of GLPCI_CNF - 0x0132

6.2.19.28.4 Data High of GLPCI_CNF - 0x0133

6.2.20 CORER Registers Auto-load Module Section Summary Table

Default setup to registers and internal memories that load on CORER events.

Word Offset	Description	Reference
0x0000	Module Length	See section 6.2.20.1
0x0001 - 0023	NVM contents for PFGEN_PORTMDIO_NUM	See section 6.2.20.2
0x0024 - 0027	NVM contents for GLINT_CTL	See section 6.2.20.3
0x002C - 0036	NVM contents for PRTDCB_TDPUC	See section 6.2.20.4
0x0037 - 0049	NVM contents for GL_SWT_L2TAGTXIB	See section 6.2.20.5
0x004A - 004B	NVM contents for GLLAN_TSOMSK_F	See section 6.2.20.6
0x004C - 004D	NVM contents for GLLAN_TSOMSK_M	See section 6.2.20.7
0x004E - 004F	NVM contents for GLLAN_TSOMSK_L	See section 6.2.20.8
0x0050 - 0054	NVM contents for GLLAN_TCTL_1	See section 6.2.20.9



Word Offset	Description	Reference
0x0057 - 0069	NVM contents for GL_SWT_L2TAGRXEB	See section 6.2.20.10
0x006E - 0078	NVM contents for PRTDCB_GENC	See section 6.2.20.11
0x0079 - 007C	NVM contents for GLDCB_GENC	See section 6.2.20.12
0x007D - 0087	NVM contents for PRT_PPRS_CTRL	See section 6.2.20.13
0x0088 - 0092	NVM contents for PRTDCB_TFLLPC	See section 6.2.20.14
0x0093 - 009D	NVM contents for PRTDCB_TDPMC	See section 6.2.20.15
0x009E - 00A8	NVM contents for PRTDCB_TFPFCC	See section 6.2.20.16
0x00AD - 00B7	NVM contents for PRTDCB_TCPMC	See section 6.2.20.17
0x00B8 - 00C2	NVM contents for PRTDCB_TCPFCC	See section 6.2.20.18
0x00C3 - 00CD	NVM contents for PRTDCB_TCPFCTCC	See section 6.2.20.19
0x00CE - 00D2	NVM contents for GLDCB_TGENC_TUPM	See section 6.2.20.20
0x00D9 - 00E3	NVM contents for PRTRPB_SLW	See section 6.2.20.22
0x00E4 - 00EE	NVM contents for PRTRPB_SPS	See section 6.2.20.22
0x00F3 - 00F7	NVM contents for GLRPB_PHW	See section 6.2.20.23
0x00F8 - 00F9	NVM contents for GLRPB_PLW	See section 6.2.20.24
0x00FA - 0104	NVM contents for PRTDCB_TCLLPC	See section 6.2.20.25
0x0111 - 0115	NVM contents for GLSCD_CREDITSQUANTA	See section 6.2.20.26
0x011E - 0140	NVM contents for GLHMC_SDPART	See section 6.2.20.27
0x0145 - 0149	NVM contents for GLLAN_TCTL_0	See section 6.2.20.28
0x014C - 0150	NVM contents for GLTLAN_MAX_TCBCMD	See section 6.2.20.29
0x0159 - 01B0	NVM contents for PRTDCB_RETSTCC	See section 6.2.20.30
0x01B1 - 01BB	NVM contents for PRTDCB_RPPMC	See section 6.2.20.31
0x01BC - 01C6	NVM contents for PRTDCB_RETSC	See section 6.2.20.32
0x01C7 - 01CB	NVM contents for GLDCB_RSPMC	See section 6.2.20.33
0x01D2 - 01D6	NVM contents for GLLAN_RCTL_1	See section 6.2.20.34
0x01D9 - 01FB	NVM contents for GLLAN_PF_RECIPCE	See section 6.2.20.35
0x01FC - 021E	NVM contents for PFLAN_QALLOC	See section 6.2.20.36
0x021F - 0241	NVM contents for PFGEN_PORTNUM	See section 6.2.20.37
0x0242 - 0264	NVM contents for PF_VT_PFALLOC	See section 6.2.20.38
0x0269 - 028B	NVM contents for GLANL_L2ULP	See section 6.2.20.39
0x028C - 029E	NVM contents for GL_SWT_L2TAGCTRL	See section 6.2.20.40
0x029F - 02C1	NVM contents for PFQF_CTL_0	See section 6.2.20.41
0x02C2 - 02CC	NVM contents for PRT_L2TAGSEN	See section 6.2.20.42
0x02CD - 02EF	NVM contents for PFQF_FDALLOC	See section 6.2.20.43
0x02F0 - 02FA	NVM contents for PRT_MSCCNT	See section 6.2.20.44
0x02FB - 0305	NVM contents for PRT_SWT_BSCCNT	See section 6.2.20.45
0x0306 - 0310	NVM contents for PRT_SWT_SCBI	See section 6.2.20.46
0x0311 - 031B	NVM contents for PRT_SWT_SCTC	See section 6.2.20.47
0x031C - 0326	NVM contents for PRTQF_CTL_0	See section 6.2.20.48



Word Offset	Description	Reference
0x0327 - 0349	NVM contents for GLQF_PE_INSET	See section 6.2.20.49
0x034A - 0350	NVM contents for GLQF_FDENA	See section 6.2.20.50
0x0361 - 0364	NVM contents for GLQF_CTL	See section 6.2.20.51
0x0381 - 038B	NVM contents for PRT_SWR_PM_THR	See section 6.2.20.52
0x038C - 03A8	NVM contents for GLQF_HKEY	See section 6.2.20.53
0x03AB	DPU_IMEM Attributes	See section 6.2.20.54
0x03AE	DPU_IMEM Data	See section 6.2.20.55
0x23B0	DPU_RECIPE_ADDRESS Attributes	See section 6.2.20.56
0x23B3	DPU_RECIPE_ADDRESS Data	See section 6.2.20.57
0x25B5	DPU_RECIPE_CAM Attributes	See section 6.2.20.58
0x25B8	DPU_RECIPE_CAM Data	See section 6.2.20.59
0x29BA	DPU_RECIPE_MASK Attributes	See section 6.2.20.60
0x29BD	DPU_RECIPE_MASK Data	See section 6.2.20.61
0x29DF	ANA_IMEM Attributes	See section 6.2.20.62
0x29E2	ANA_IMEM Data	See section 6.2.20.63
0x2A84	ANA_NH Attributes	See section 6.2.20.64
0x2A87	ANA_NH Data	See section 6.2.20.65
0x2AD9	ANA_SKIP Attributes	See section 6.2.20.66
0x2ADC	ANA_SKIP Data	See section 6.2.20.67
0x2B2E	ANA_REPLACE Attributes	See section 6.2.20.68
0x2B31	ANA_REPLACE Data	See section 6.2.20.69
0x2B83	ANA_MERGE Attributes	See section 6.2.20.70
0x2B86	ANA_MERGE Data	See section 6.2.20.71

6.2.20.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		Length in: 2 bytes unit - 1 First Section -> Word: CORER Registers Auto-load Module -> Module Length Last Section -> Word: CORER Registers Auto-load Module -> ANA_MERGE Data

6.2.20.2 PFGEN_PORTMDIO_NUM - (0x0001 - 0023)

6.2.20.2.1 Starting Address Low at PFGEN_PORTMDIO_NUM - 0x0001

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFGEN_PORTMDIO_NUM for PF[0]	0x3F100	
3:0	Type	0x2	



6.2.20.2.2 Starting Address High at PFGEN_PORTMDIO_NUM - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFGEN_PORTMDIO_NUM for PF[0]		

6.2.20.2.3 Attributes at PFGEN_PORTMDIO_NUM - 0x0003

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.2.4 Data Low of PFGEN_PORTMDIO_NUM[PF] - 0x0004 + 2*PF, PF=0...15

6.2.20.2.5 Data High of PFGEN_PORTMDIO_NUM[PF] - 0x0005 + 2*PF, PF=0...15

6.2.20.3 GLINT_CTL - (0x0024 - 0027)

6.2.20.3.1 Address Low at GLINT_CTL - 0x0024

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLINT_CTL	0x3F800	
3:0	Type	0x1	

6.2.20.3.2 Address High at GLINT_CTL - 0x0025

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLINT_CTL		

6.2.20.3.3 Data Low of GLINT_CTL - 0x0026

6.2.20.3.4 Data High of GLINT_CTL - 0x0027

6.2.20.4 PRTDCB_TDPUC - (0x002C - 0036)

6.2.20.4.1 Starting Address Low at PRTDCB_TDPUC - 0x002C

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_TDPUC for PRT[0]	0x44100	
3:0	Type	0x2	



6.2.20.4.2 Starting Address High at PRTDCB_TDPUC - 0x002D

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_TDPUC for PRT[0]		

6.2.20.4.3 Attributes at PRTDCB_TDPUC - 0x002E

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.5 GL_SWT_L2TAGTXIB - (0x0037 - 0049)

6.2.20.5.1 Starting Address Low at GL_SWT_L2TAGTXIB - 0x0037

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GL_SWT_L2TAGTXIB	0x442B8	
3:0	Type	0x2	

6.2.20.5.2 Starting Address High at GL_SWT_L2TAGTXIB - 0x0038

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GL_SWT_L2TAGTXIB		

6.2.20.5.3 Attributes at GL_SWT_L2TAGTXIB - 0x0039

Bits	Field Name	Default NVM Value	Description
15:5	Length	0xB	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.6 GLLAN_TSOMSK_F - (0x004A - 004B)

6.2.20.6.1 Data Low of GLLAN_TSOMSK_F - 0x004A

6.2.20.6.2 Data High of GLLAN_TSOMSK_F - 0x004B

6.2.20.7 GLLAN_TSOMSK_M - (0x004C - 004D)

6.2.20.7.1 Data Low of GLLAN_TSOMSK_M - 0x004C



6.2.20.7.2 Data High of GLLAN_TSOMSK_M - 0x004D

6.2.20.8 GLLAN_TSOMSK_L - (0x004E - 004F)

6.2.20.8.1 Data Low of GLLAN_TSOMSK_L - 0x004E

6.2.20.8.2 Data High of GLLAN_TSOMSK_L - 0x004F

6.2.20.9 GLLAN_TCTL_1 - (0x0050 - 0054)

6.2.20.9.1 Starting Address Low at GLLAN_TCTL_1 - 0x0050

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLLAN_TCTL_1	0x442F0	
3:0	Type	0x2	

6.2.20.9.2 Starting Address High at GLLAN_TCTL_1 - 0x0051

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLLAN_TCTL_1		

6.2.20.9.3 Attributes at GLLAN_TCTL_1 - 0x0052

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.10 GL_SWT_L2TAGRXEB - (0x0057 - 0069)

6.2.20.10.1 Starting Address Low at GL_SWT_L2TAGRXEB - 0x0057

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GL_SWT_L2TAGRXEB	0x51000	
3:0	Type	0x2	

6.2.20.10.2 Starting Address High at GL_SWT_L2TAGRXEB - 0x0058

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GL_SWT_L2TAGRXEB		



6.2.20.10.3 Attributes at GL_SWT_L2TAGRXEB - 0x0059

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x8	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.11 PRTDCB_GENC - (0x006E - 0078)

6.2.20.11.1 Starting Address Low at PRTDCB_GENC - 0x006E

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_GENC for PRT[0]	0x83000	
3:0	Type	0x2	

6.2.20.11.2 Starting Address High at PRTDCB_GENC - 0x006F

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_GENC for PRT[0]		

6.2.20.11.3 Attributes at PRTDCB_GENC - 0x0070

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.11.4 Data Low of PRTDCB_GENC[PRT] - $0x0071 + 2 * PRT$, PRT=0...3

6.2.20.11.5 Data High of PRTDCB_GENC[PRT] - $0x0072 + 2 * PRT$, PRT=0...3

6.2.20.12 GLDCB_GENC - (0x0079 - 007C)

6.2.20.12.1 Address Low at GLDCB_GENC - 0x0079

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLDCB_GENC	0x83044	
3:0	Type	0x1	



6.2.20.12.2 Address High at GLDCB_GENC - 0x007A

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLDCB_GENC		

6.2.20.12.3 Data Low of GLDCB_GENC - 0x007B

6.2.20.12.4 Data High of GLDCB_GENC - 0x007C

6.2.20.13 PRT_PPRS_CTRL - (0x007D - 0087)

6.2.20.13.1 Starting Address Low at PRT_PPRS_CTRL - 0x007D

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRT_PPRS_CTRL for PRT[0]	0x86000	
3:0	Type	0x2	

6.2.20.13.2 Starting Address High at PRT_PPRS_CTRL - 0x007E

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRT_PPRS_CTRL for PRT[0]		

6.2.20.13.3 Attributes at PRT_PPRS_CTRL - 0x007F

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.14 PRTDCB_TFLLPC - (0x0088 - 0092)

6.2.20.14.1 Starting Address Low at PRTDCB_TFLLPC - 0x0088

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_TFLLPC for PRT[0]	0x98000	
3:0	Type	0x2	



6.2.20.14.2 Starting Address High at PRTDCB_TFLLPC - 0x0089

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_TFLLPC for PRT[0]		

6.2.20.14.3 Attributes at PRTDCB_TFLLPC - 0x008A

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.15 PRTDCB_TDPMC - (0x0093 - 009D)

6.2.20.15.1 Starting Address Low at PRTDCB_TDPMC - 0x0093

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_TDPMC for PRT[0]	0xA0180	
3:0	Type	0x2	

6.2.20.15.2 Starting Address High at PRTDCB_TDPMC - 0x0094

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_TDPMC for PRT[0]		

6.2.20.15.3 Attributes at PRTDCB_TDPMC - 0x0095

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.15.4 Data Low of PRTDCB_TDPMC[PRT] - 0x0096 + 2*PRT, PRT=0...3

6.2.20.15.5 Data High of PRTDCB_TDPMC[PRT] - 0x0097 + 2*PRT, PRT=0...3



6.2.20.16 PRTDCB_TFPFCC - (0x009E - 00A8)

6.2.20.16.1 Starting Address Low at PRTDCB_TFPFCC - 0x009E

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_TFPFCC for PRT[0]	0xA01A0	
3:0	Type	0x2	

6.2.20.16.2 Starting Address High at PRTDCB_TFPFCC - 0x009F

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_TFPFCC for PRT[0]		

6.2.20.16.3 Attributes at PRTDCB_TFPFCC - 0x00A0

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.17 PRTDCB_TCPMC - (0x00AD - 00B7)

6.2.20.17.1 Starting Address Low at PRTDCB_TCPMC - 0x00AD

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_TCPMC for PRT[0]	0xA21A0	
3:0	Type	0x2	

6.2.20.17.2 Starting Address High at PRTDCB_TCPMC - 0x00AE

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_TCPMC for PRT[0]		

6.2.20.17.3 Attributes at PRTDCB_TCPMC - 0x00AF0

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	



Bits	Field Name	Default NVM Value	Description
2:0	Width	0x0	

6.2.20.17.4 Data Low of PRTDCB_TCPMC[PRT] - 0x00B0 + 2*PRT, PRT=0...3

6.2.20.17.5 Data High of PRTDCB_TCPMC[PRT] - 0x00B1 + 2*PRT, PRT=0...3

6.2.20.18 PRTDCB_TCPFCPC - (0x00B8 - 00C2)

6.2.20.18.1 Starting Address Low at PRTDCB_TCPFCPC - 0x00B8

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_TCPFCPC for PRT[0]	0xA21C0	
3:0	Type	0x2	

6.2.20.18.2 Starting Address High at PRTDCB_TCPFCPC - 0x00B9

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_TCPFCPC for PRT[0]		

6.2.20.18.3 Attributes at PRTDCB_TCPFCPC - 0x00BA

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.19 PRTDCB_TCPFCTCC - (0x00C3 - 00CD)

6.2.20.19.1 Starting Address Low at PRTDCB_TCPFCTCC - 0x00C3

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_TCPFCTCC for PRT[0]	0xA21E0	
3:0	Type	0x2	



6.2.20.19.2 Starting Address High at PRTDCB_TCPFCTCC - 0x00C4

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_TCPFCTCC for PRT[0]		

6.2.20.19.3 Attributes at PRTDCB_TCPFCTCC - 0x00C5

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.20 GLDCB_TGENC_TUPM - (0x00CE - 00D2)

6.2.20.20.1 Starting Address Low at GLDCB_TGENC_TUPM - 0x00CE

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLDCB_TGENC_TUPM	0xA2200	
3:0	Type	0x2	

6.2.20.20.2 Starting Address High at GLDCB_TGENC_TUPM - 0x00CF

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLDCB_TGENC_TUPM		

6.2.20.20.3 Attributes at GLDCB_TGENC_TUPM - 0x00D0

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.21 PRTRPB_SLW- (0x00D9 - 00E3)

6.2.20.21.1 Starting Address Low at PRTRPB_SLW - 0x00D9

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTRPB_SLW for PRT[0]	0xAC6A0	



Bits	Field Name	Default NVM Value	Description
3:0	Type	0x2	

6.2.20.21.2 Starting Address High at PRTRPB_SLW - 0x00DA

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTRPB_SLW for PRT[0]		

6.2.20.21.3 Attributes at PRTRPB_SLW - 0x00DB

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.21.4 Data Low of PRTRPB_SLW[PRT] - 0x00DC + 2*PRT, PRT=0...3

6.2.20.21.5 Data High of PRTRPB_SLW[PRT] - 0x00DD + 2*PRT, PRT=0...3

6.2.20.22 PRTRPB_SPS - (0x00E4 - 00EE)

6.2.20.22.1 Starting Address Low at PRTRPB_SPS - 0x00E4

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTRPB_SPS for PRT[0]	0xAC7C0	
3:0	Type	0x2	

6.2.20.22.2 Starting Address High at PRTRPB_SPS - 0x00E5

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTRPB_SPS for PRT[0]		

6.2.20.22.3 Attributes at PRTRPB_SPS - 0x00E6

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.20.22.4 Data Low of PRTRPB_SPS[PRT] - 0x00E7 + 2*PRT, PRT=0...3

6.2.20.22.5 Data High of PRTRPB_SPS[PRT] - 0x00E8 + 2*PRT, PRT=0...3

6.2.20.23 GLRPB_PHW - (0x00F3 - 00F7)

6.2.20.23.1 Starting Address Low at GLRPB_PHW - 0x00F3

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLRPB_PHW	0xAC844	
3:0	Type	0x2	

6.2.20.23.2 Starting Address High at GLRPB_PHW - 0x00F4

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLRPB_PHW		

6.2.20.23.3 Attributes at GLRPB_PHW - 0x00F5

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.23.4 Data Low of GLRPB_PHW - 0x00F6

6.2.20.23.5 Data High of GLRPB_PHW - 0x00F7

6.2.20.24 GLRPB_PLW - (0x00F8 - 00F9)

6.2.20.24.1 Data Low of GLRPB_PLW - 0x00F8

6.2.20.24.2 Data High of GLRPB_PLW - 0x00F9

6.2.20.25 PRTDCB_TCLLPC - (0x00FA - 0104)

6.2.20.25.1 Starting Address Low at PRTDCB_TCLLPC - 0x00FA

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_TCLLPC for PRT[0]	0xAE000	
3:0	Type	0x2	



6.2.20.25.2 Starting Address High at PRTDCB_TCLLPC - 0x00FB

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_TCLLPC for PRT[0]		

6.2.20.25.3 Attributes at PRTDCB_TCLLPC - 0x00FC

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.26 GLSCD_CREDITSPERQUANTA - (0x0111 - 0115)

6.2.20.26.1 Starting Address Low at GLSCD_CREDITSPERQUANTA - 0x0111

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLSCD_CREDITSPERQUANTA	0xB2144	
3:0	Type	0x2	

6.2.20.26.2 Starting Address High at GLSCD_CREDITSPERQUANTA - 0x0112

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLSCD_CREDITSPERQUANTA		

6.2.20.26.3 Attributes at GLSCD_CREDITSPERQUANTA - 0x0113

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x3	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.27 GLHMC_SDPART - (0x011E - 0140)

6.2.20.27.1 Starting Address Low at GLHMC_SDPART - 0x011E

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLHMC_SDPART	0xC0800	
3:0	Type	0x2	



6.2.20.27.2 Starting Address High at GLHMC_SDPART - 0x011F

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLHMC_SDPART		

6.2.20.27.3 Attributes at GLHMC_SDPART - 0x0120

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.27.4 Data Low of GLHMC_SDPART[n] - 0x0121 + 2*n, n=0...15

6.2.20.27.5 Data High of GLHMC_SDPART[n] - 0x0122 + 2*n, n=0...15

6.2.20.28 GLLAN_TCTL_0 - (0x0145 - 0149)

6.2.20.28.1 Starting Address Low at GLLAN_TCTL_0 - 0x0145

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLLAN_TCTL_0	0xE6488	
3:0	Type	0x2	

6.2.20.28.2 Starting Address High at GLLAN_TCTL_0 - 0x0146

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLLAN_TCTL_0		

6.2.20.28.3 Attributes at GLLAN_TCTL_0 - 0x0147

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.20.29 GLTLAN_MAX_TCBCMD - (0x014C - 0150)

6.2.20.29.1 Starting Address Low at GLTLAN_MAX_TCBCMD - 0x014C

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLTLAN_MAX_TCBCMD	0xE64D4	
3:0	Type	0x2	

6.2.20.29.2 Starting Address High at GLTLAN_MAX_TCBCMD - 0x014D

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLTLAN_MAX_TCBCMD		

6.2.20.29.3 Attributes at GLTLAN_MAX_TCBCMD - 0x014E

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x3	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.30 PRTDCB_RETSTCC - (0x0159 - 01B0)

6.2.20.30.1 Starting Address Low at PRTDCB_RETSTCC - 0x0159

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_RETSTCC for PRT[0]	0x122180	
3:0	Type	0x2	

6.2.20.30.2 Starting Address High at PRTDCB_RETSTCC - 0x015A

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_RETSTCC for PRT[0]		

6.2.20.30.3 Attributes at PRTDCB_RETSTCC - 0x015B

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	



Bits	Field Name	Default NVM Value	Description
2:0	Width	0x0	

6.2.20.30.4 Data Low of PRTDCB_RETSTCC[n,PRT] - $0x015C + 11*n + 2*PRT$, n=0...7, PRT=0...3

6.2.20.30.5 Data High of PRTDCB_RETSTCC[n,PRT] - $0x015D + 11*n + 2*PRT$, n=0...7, PRT=0...3

6.2.20.31 PRTDCB_RPPMC - (0x01B1 - 01BB)

6.2.20.31.1 Starting Address Low at PRTDCB_RPPMC - 0x01B1

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_RPPMC for PRT[0]	0x1223A0	
3:0	Type	0x2	

6.2.20.31.2 Starting Address High at PRTDCB_RPPMC - 0x01B2

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_RPPMC for PRT[0]		

6.2.20.31.3 Attributes at PRTDCB_RPPMC - 0x01B3

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.31.4 Data High of PRTDCB_RPPMC[PRT] - $0x01B5 + 2*PRT$, PRT=0...3

6.2.20.32 PRTDCB_RETSC - (0x01BC - 01C6)

6.2.20.32.1 Starting Address Low at PRTDCB_RETSC - 0x01BC

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_RETSC for PRT[0]	0x1223E0	
3:0	Type	0x2	



6.2.20.32.2 Starting Address High at PRTDCB_RETSC - 0x01BD

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_RETSC for PRT[0]		

6.2.20.32.3 Attributes at PRTDCB_RETSC - 0x01BE

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.32.4 Data Low of PRTDCB_RETSC[PRT] - 0x01BF + 2*PRT, PRT=0...3

6.2.20.32.5 Data High of PRTDCB_RETSC[PRT] - 0x01C0 + 2*PRT, PRT=0...3

6.2.20.33 GLDCB_RSPMC - (0x01C7 - 01CB)

6.2.20.33.1 Starting Address Low at GLDCB_RSPMC - 0x01C7

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLDCB_RSPMC	0x122604	
3:0	Type	0x2	

6.2.20.33.2 Starting Address High at GLDCB_RSPMC - 0x01C8

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLDCB_RSPMC		

6.2.20.33.3 Attributes at GLDCB_RSPMC - 0x01C9

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.20.34 GLLAN_RCTL_1 - (0x01D2 - 01D6)

6.2.20.34.1 Starting Address Low at GLLAN_RCTL_1 - 0x01D2

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLLAN_RCTL_1	0x12A504	
3:0	Type	0x2	

6.2.20.34.2 Starting Address High at GLLAN_RCTL_1 - 0x01D3

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLLAN_RCTL_1		

6.2.20.34.3 Attributes at GLLAN_RCTL_1 - 0x01D4

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.35 GLLAN_PF_RECIPE - (0x01D9 - 01FB)

6.2.20.35.1 Starting Address Low at GLLAN_PF_RECIPE - 0x01D9

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLLAN_PF_RECIPE	0x12A5E0	
3:0	Type	0x2	

6.2.20.35.2 Starting Address High at GLLAN_PF_RECIPE - 0x01DA

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLLAN_PF_RECIPE		

6.2.20.35.3 Attributes at GLLAN_PF_RECIPE - 0x01DB

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.20.36 PFLAN_QALLOC - (0x01FC - 021E)

6.2.20.36.1 Starting Address Low at PFLAN_QALLOC - 0x01FC

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFLAN_QALLOC for PF[0]	0x1C0400	
3:0	Type	0x2	

6.2.20.36.2 Starting Address High at PFLAN_QALLOC - 0x01FD

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFLAN_QALLOC for PF[0]		

6.2.20.36.3 Attributes at PFLAN_QALLOC - 0x01FE0

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.36.4 Data Low of PFLAN_QALLOC[PF] - $0x01FF + 2*PF$, PF=0...15

6.2.20.36.5 Data High of PFLAN_QALLOC[PF] - $0x0200 + 2*PF$, PF=0...15

6.2.20.37 PFGEN_PORTNUM - (0x021F - 0241)

6.2.20.37.1 Starting Address Low at PFGEN_PORTNUM - 0x021F

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFGEN_PORTNUM for PF[0]	0x1C0480	
3:0	Type	0x2	

6.2.20.37.2 Starting Address High at PFGEN_PORTNUM - 0x0220

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFGEN_PORTNUM for PF[0]		



6.2.20.37.3 Attributes at PFGEN_PORTNUM - 0x0221

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.37.4 Data Low of PFGEN_PORTNUM[PF] - 0x0222 + 2*PF, PF=0...15

6.2.20.37.5 Data High of PFGEN_PORTNUM[PF] - 0x0223 + 2*PF, PF=0...15

6.2.20.38 PF_VT_PFALLOC - (0x0242 - 0264)

6.2.20.38.1 Starting Address Low at PF_VT_PFALLOC - 0x0242

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PF_VT_PFALLOC for PF[0]	0x1C0500	
3:0	Type	0x2	

6.2.20.38.2 Starting Address High at PF_VT_PFALLOC - 0x0243

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PF_VT_PFALLOC for PF[0]		

6.2.20.38.3 Attributes at PF_VT_PFALLOC - 0x0244

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.38.4 Data Low of PF_VT_PFALLOC[PF] - 0x0245 + 2*PF, PF=0...15

6.2.20.38.5 Data High of PF_VT_PFALLOC[PF] - 0x0246 + 2*PF, PF=0...15

6.2.20.39 GLANL_L2ULP - (0x0269 - 028B)

6.2.20.39.1 Starting Address Low at GLANL_L2ULP - 0x0269

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLANL_L2ULP	0x1C0A2C	



Bits	Field Name	Default NVM Value	Description
3:0	Type	0x2	

6.2.20.39.2 Starting Address High at GLANL_L2ULP - 0x026A

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLANL_L2ULP		

6.2.20.39.3 Attributes at GLANL_L2ULP - 0x026B

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.40 GL_SWT_L2TAGCTRL - (0x028C - 029E)

6.2.20.40.1 Starting Address Low at GL_SWT_L2TAGCTRL - 0x028C

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GL_SWT_L2TAGCTRL	0x1C0A70	
3:0	Type	0x2	

6.2.20.40.2 Starting Address High at GL_SWT_L2TAGCTRL - 0x028D

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GL_SWT_L2TAGCTRL		

6.2.20.40.3 Attributes at GL_SWT_L2TAGCTRL - 0x028E

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x8	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.40.4 Data Low of GL_SWT_L2TAGCTRL[n] - 0x028F + 2*n, n=0...7

6.2.20.40.5 Data High of GL_SWT_L2TAGCTRL[n] - 0x0290 + 2*n, n=0...7



6.2.20.41 PFQF_CTL_0 - (0x029F - 02C1)

6.2.20.41.1 Starting Address Low at PFQF_CTL_0 - 0x029F

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFQF_CTL_0 for PF[0]	0x1C0AC0	
3:0	Type	0x2	

6.2.20.41.2 Starting Address High at PFQF_CTL_0 - 0x02A0

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFQF_CTL_0 for PF[0]		

6.2.20.41.3 Attributes at PFQF_CTL_0 - 0x02A1

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.41.4 Data Low of PFQF_CTL_0[PF] - 0x02A2 + 2*PF, PF=0...15

6.2.20.41.5 Data High of PFQF_CTL_0[PF] - 0x02A3 + 2*PF, PF=0...15

6.2.20.42 PRT_L2TAGSEN - (0x02C2 - 02CC)

6.2.20.42.1 Starting Address Low at PRT_L2TAGSEN - 0x02C2

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRT_L2TAGSEN for PRT[0]	0x1C0B20	
3:0	Type	0x2	

6.2.20.42.2 Starting Address High at PRT_L2TAGSEN - 0x02C3

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRT_L2TAGSEN for PRT[0]		



6.2.20.42.3 Attributes at PRT_L2TAGSEN - 0x02C4

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.42.4 Data Low of PRT_L2TAGSEN[PRT] - 0x02C5 + 2*PRT, PRT=0...3

6.2.20.42.5 Data High of PRT_L2TAGSEN[PRT] - 0x02C6 + 2*PRT, PRT=0...3

6.2.20.43 PFQF_FDALLOC - (0x02CD - 02EF)

6.2.20.43.1 Starting Address Low at PFQF_FDALLOC - 0x02CD

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PFQF_FDALLOC for PF[0]	0x246280	
3:0	Type	0x2	

6.2.20.43.2 Starting Address High at PFQF_FDALLOC- 0x02CE

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PFQF_FDALLOC for PF[0]		

6.2.20.43.3 Attributes at PFQF_FDALLOC - 0x02CF

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.43.4 Data Low of PFQF_FDALLOC[PF] - 0x02D0 + 2*PF, PF=0...15

6.2.20.43.5 Data High of PFQF_FDALLOC[PF] - 0x02D1 + 2*PF, PF=0...15



6.2.20.44 PRT_MSCCNT - (0x02F0 - 02FA)

6.2.20.44.1 Starting Address Low at PRT_MSCCNT - 0x02F0

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRT_MSCCNT for PRT[0]	0x256BA0	
3:0	Type	0x2	

6.2.20.44.2 Starting Address High at PRT_MSCCNT - 0x02F1

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRT_MSCCNT for PRT[0]		

6.2.20.44.3 Attributes at PRT_MSCCNT - 0x02F2

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.45 PRT_SWT_BSCCNT - (0x02FB - 0305)

6.2.20.45.1 Starting Address Low at PRT_SWT_BSCCNT - 0x02FB

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRT_SWT_BSCCNT for PRT[0]	0x256C60	
3:0	Type	0x2	

6.2.20.45.2 Starting Address High at PRT_SWT_BSCCNT - 0x02FC

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRT_SWT_BSCCNT for PRT[0]		

6.2.20.45.3 Attributes at PRT_SWT_BSCCNT - 0x02FD

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	



Bits	Field Name	Default NVM Value	Description
2:0	Width	0x0	

6.2.20.46 PRT_SWT_SCBI - (0x0306 - 0310)

6.2.20.46.1 Starting Address Low at PRT_SWT_SCBI - 0x0306

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRT_SWT_SCBI for PRT[0]	0x256D60	
3:0	Type	0x2	

6.2.20.46.2 Starting Address High at PRT_SWT_SCBI - 0x0307

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRT_SWT_SCBI for PRT[0]		

6.2.20.46.3 Attributes at PRT_SWT_SCBI - 0x0308

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.47 PRT_SWT_SCTC - (0x0311 - 031B)

6.2.20.47.1 Starting Address Low at PRT_SWT_SCTC - 0x0311

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRT_SWT_SCTC for PRT[0]	0x256DE0	
3:0	Type	0x2	

6.2.20.47.2 Starting Address High at PRT_SWT_SCTC - 0x0312

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRT_SWT_SCTC for PRT[0]		



6.2.20.47.3 Attributes at PRT_SWT_SCTC - 0x03130

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.48 PRTQF_CTL_0 - (0x031C - 0326)

6.2.20.48.1 Starting Address Low at PRTQF_CTL_0 - 0x031C

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTQF_CTL_0 for PRT[0]	0x256E60	
3:0	Type	0x2	

6.2.20.48.2 Starting Address High at PRTQF_CTL_0 - 0x031D

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTQF_CTL_0 for PRT[0]		

6.2.20.48.3 Attributes at PRTQF_CTL_0 - 0x031E

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.48.4 Data Low of PRTQF_CTL_0[PRT] - 0x031F + 2*PRT, PRT=0...3

6.2.20.48.5 Data High of PRTQF_CTL_0[PRT] - 0x0320 + 2*PRT, PRT=0...3

6.2.20.49 GLQF_PE_INSET - (0x0327 - 0349)

6.2.20.49.1 Starting Address Low at GLQF_PE_INSET - 0x0327

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLQF_PE_INSET	0x269140	
3:0	Type	0x2	



6.2.20.49.2 Starting Address High at GLQF_PE_INSET - 0x0328

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLQF_PE_INSET		

6.2.20.49.3 Attributes at GLQF_PE_INSET - 0x0329

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x10	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.50 GLQF_FDENA - (0x034A - 0350)

6.2.20.50.1 Starting Address Low at GLQF_FDENA - 0x034A

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLQF_FDENA	0x2698A8	
3:0	Type	0x2	

6.2.20.50.2 Starting Address High at GLQF_FDENA - 0x034B

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLQF_FDENA		

6.2.20.50.3 Attributes at GLQF_FDENA - 0x034C

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.51 GLQF_CTL - (0x0361 - 0364)

6.2.20.51.1 Address Low at GLQF_CTL - 0x0361

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLQF_CTL	0x269BA4	
3:0	Type	0x1	



6.2.20.51.2 Address High at GLQF_CTL - 0x0362

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLQF_CTL		

6.2.20.51.3 Data Low of GLQF_CTL - 0x0363

6.2.20.51.4 Data High of GLQF_CTL - 0x0364

6.2.20.52 PRT_SWR_PM_THR - (0x0381 - 038B)

6.2.20.52.1 Starting Address Low at PRT_SWR_PM_THR - 0x0381

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRT_SWR_PM_THR for PRT[0]	0x26CD00	
3:0	Type	0x2	

6.2.20.52.2 Starting Address High at PRT_SWR_PM_THR - 0x0382

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRT_SWR_PM_THR for PRT[0]		

6.2.20.52.3 Attributes at PRT_SWR_PM_THR - 0x0383

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.53 GLQF_HKEY - (0x038C - 03A8)

6.2.20.53.1 Starting Address Low at GLQF_HKEY - 0x038C

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLQF_HKEY	0x270140	
3:0	Type	0x2	



6.2.20.53.2 Starting Address High at GLQF_HKEY - 0x038D

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLQF_HKEY		

6.2.20.53.3 Attributes at GLQF_HKEY - 0x038E

Bits	Field Name	Default NVM Value	Description
15:5	Length	0xD	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.20.53.4 Data Low of GLQF_HKEY[n] - 0x038F + 2*n, n=0...12

6.2.20.53.5 Data High of GLQF_HKEY[n] - 0x0390 + 2*n, n=0...12

6.2.20.54 DPU_IMEM Attributes - 0x03AB

Part of a Type 3 structure to load the DPU_IMEM.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x400	
4:3	Skip	0x00	
2:0	Width	0x2	

6.2.20.55 DPU_IMEM Data - 0x03AE

Raw data module length: 8192 words.

Part of a Type 3 structure to load the DPU_IMEM.

6.2.20.56 DPU_RECIPE_ADDRESS Attributes - 0x23B0

Part of a Type 3 structure to load the DPU_RECIPE_ADDRESS.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x100	
4:3	Skip	0x00	

6.2.20.57 DPU_RECIPE_ADDRESS Data - 0x23B3

Raw data module length: 512 words.

Part of a Type 3 structure to load the DPU_RECIPE_ADDRESS.



6.2.20.58 DPU_RECIPES_CAM Attributes - 0x25B5

Part of a Type 3 structure to load the DPU_RECIPES_CAM.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x100	
4:3	Skip	0x00	

6.2.20.59 DPU_RECIPES_CAM Data - 0x25B8

Raw data module length: 1024 words.

Part of a Type 3 structure to load the DPU_RECIPES_CAM.

6.2.20.60 DPU_RECIPES_MASK Attributes - 0x29BA

Part of a Type 3 structure to load the DPU_RECIPES_MASK.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x008	
4:3	Skip	0x00	

6.2.20.61 DPU_RECIPES_MASK Data - 0x29BD

Raw data module length: 32 words.

Part of a Type 3 structure to load the DPU_RECIPES_MASK.

6.2.20.62 ANA_IMEM Attributes - 0x29DF

Part of a Type 3 structure to load the DPU_IMEM.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x28	
4:3	Skip	0x00	

6.2.20.63 ANA_IMEM Data - 0x29E2

Raw data module length: 160 words.

Part of a Type 3 structure to load the DPU_IMEM.

6.2.20.64 ANA_NH Attributes - 0x2A84

Part of a Type 3 structure to load the DPU_IMEM.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x28	



Bits	Field Name	Default NVM Value	Description
4:3	Skip	0x00	

6.2.20.65 ANA_NH Data - 0x2A87

Raw data module length: 80 words.

Part of a Type 3 structure to load the DPU_IMEM.

6.2.20.66 ANA_SKIP Attributes - 0x2AD9

Part of a Type 3 structure to load the DPU_IMEM.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x28	
4:3	Skip	0x00	

6.2.20.67 ANA_SKIP Data - 0x2ADC

Raw data module length: 80 words.

Part of a Type 3 structure to load the DPU_IMEM.

6.2.20.68 ANA_REPLACE Attributes - 0x2B2E

Part of a Type 3 structure to load the DPU_IMEM.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x28	
4:3	Skip	0x00	

6.2.20.69 ANA_REPLACE Data - 0x2B31

Raw data module length: 80 words.

Part of a Type 3 structure to load the DPU_IMEM.

6.2.20.70 ANA_MERGE Attributes - 0x2B83

Part of a Type 3 structure to load the DPU_IMEM.

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x28	
4:3	Skip	0x00	

6.2.20.71 ANA_MERGE Data - 0x2B86

Raw data module length: 80 words.



Part of a Type 3 structure to load the DPU_IMEM.

6.2.21 GLOBR Registers Auto-load Module Section Summary Table

Default setup to registers that load on GLOBR events.

Word Offset	Description	Reference
0x0000	Module Length	See section 6.2.21.1
0x000C - 0016	NVM contents for PRTMAC_PCS_LINK_CTRL	See section 6.2.21.2
0x0017 - 001B	NVM contents for PRTMAC_PCS_XAUI_SWAP_A	See section 6.2.21.3
0x001C - 001D	NVM contents for PRTMAC_PCS_XAUI_SWAP_B	See section 6.2.21.4
0x001E - 0028	NVM contents for PRTMAC_HLCTL	See section 6.2.21.5
0x0029 - 0033	NVM contents for PRTMAC_PAP	See section 6.2.21.6
0x0034 - 003E	NVM contents for PRTGL_SAL	See section 6.2.21.7
0x003F - 0049	NVM contents for PRTGL_SAH	See section 6.2.21.8
0x004A - 0054	NVM contents for PRDTCB_MFLCN	See section 6.2.21.9
0x0055 - 005F	NVM contents for PRTMAC_MACC	See section 6.2.21.10
0x0060 - 0066	NVM contents for PRTMAC_HSEC_CTL_RX_MAX_PACKET_LEN	See section 6.2.21.11
0x0067 - 006D	NVM contents for PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE	See section 6.2.21.12
0x006E - 0074	NVM contents for PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE	See section 6.2.21.13
0x0075 - 007B	NVM contents for PRTMAC_HSEC_CTL_RX_ENABLE_GCP	See section 6.2.21.14
0x007C - 0082	NVM contents for PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GCP	See section 6.2.21.15
0x0083 - 0089	NVM contents for PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1	See section 6.2.21.16
0x008A - 0090	NVM contents for PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2	See section 6.2.21.17
0x0091 - 0097	NVM contents for PRTMAC_HSEC_CTL_RX_CHECK_SA_GCP	See section 6.2.21.18
0x0098 - 009E	NVM contents for PRTMAC_HSEC_CTL_RX_ENABLE_GPP	See section 6.2.21.19
0x009F - 00A5	NVM contents for PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GPP	See section 6.2.21.20
0x00A6 - 00AC	NVM contents for PRTMAC_HSEC_CTL_RX_ENABLE_PPP	See section 6.2.21.21
0x00AD - 00B3	NVM contents for PRTMAC_HSEC_CTL_RX_CHECK_UCAST_PPP	See section 6.2.21.22
0x00B4 - 00BA	NVM contents for PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL	See section 6.2.21.23
0x00BB - 00F9	NVM contents for PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA	See section 6.2.21.24
0x00FA - 0138	NVM contents for PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER	See section 6.2.21.25
0x0139 - 013F	NVM contents for PRTMAC_HSEC_CTL_TX_SA_PART1	See section 6.2.21.26
0x0140 - 0146	NVM contents for PRTMAC_HSEC_CTL_TX_SA_PART2	See section 6.2.21.27
0x0147 - 014D	NVM contents for PRTMAC_HSEC_CTL_INTERNAL	See section 6.2.21.28
0x0152 - 0158	NVM contents for PRTMAC_HSEC_CTL_XLGMII	See section 6.2.21.29
0x0159 - 015F	NVM contents for PRTMAC_HSECTL1	See section 6.2.21.30
0x0160 - 016A	NVM contents for PRRTSYN_CTL0	See section 6.2.21.31
0x016B - 016F	NVM contents for GLPM_EEE_SU	See section 6.2.21.32
0x0172 - 017C	NVM contents for PRTPM_EEER	See section 6.2.21.33
0x017D - 0187	NVM contents for PRTPM_EEEC	See section 6.2.21.34



Word Offset	Description	Reference
0x0188 - 01B3	NVM contents for PRTDCB_FCTTVN	See section 6.2.21.35
0x01B4 - 01BE	NVM contents for PRTDCB_FCRTV	See section 6.2.21.36
0x01BF - 01C9	NVM contents for PRTDCB_FCCFG	See section 6.2.21.37
0x01CA - 01D4	NVM contents for PRTMAC_HLCTLA	See section 6.2.21.38

6.2.21.1 Module Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		Length in: 2 bytes unit - 1 First Section -> Word: GLOBR Registers Auto-load Module -> Module Length Last Section -> Word: GLOBR Registers Auto-load Module -> Starting Address Low at PRTMAC_HLCTLA, for PRT[0]

6.2.21.2 PRTMAC_PCS_LINK_CTRL - (0x000C - 0016)

6.2.21.2.1 Starting Address Low at PRTMAC_PCS_LINK_CTRL - 0x000C

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_PCS_LINK_CTRL for PRT[0]	0x8C260	
3:0	Type	0x2	

6.2.21.2.2 Starting Address High at PRTMAC_PCS_LINK_CTRL - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_PCS_LINK_CTRL for PRT[0]		

6.2.21.2.3 Attributes at PRTMAC_PCS_LINK_CTRL - 0x000E

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.21.3 PRTMAC_PCS_XAUI_SWAP_A - (0x0017 - 001B)

6.2.21.3.1 Starting Address Low at PRTMAC_PCS_XAUI_SWAP_A - 0x0017

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_PCS_XAUI_SWAP_A	0x8C480	
3:0	Type	0x2	

6.2.21.3.2 Starting Address High at PRTMAC_PCS_XAUI_SWAP_A - 0x0018

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_PCS_XAUI_SWAP_A		

6.2.21.3.3 Attributes at PRTMAC_PCS_XAUI_SWAP_A - 0x0019

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.3.4 Data Low of PRTMAC_PCS_XAUI_SWAP_A - 0x001A

6.2.21.3.5 Data High of PRTMAC_PCS_XAUI_SWAP_A - 0x001B

6.2.21.4 PRTMAC_PCS_XAUI_SWAP_B - (0x001C - 001D)

6.2.21.4.1 Data Low of PRTMAC_PCS_XAUI_SWAP_B - 0x001C

6.2.21.4.2 Data High of PRTMAC_PCS_XAUI_SWAP_B - 0x001D

6.2.21.5 PRTMAC_HLCTL - (0x001E - 0028)

6.2.21.5.1 Starting Address Low at PRTMAC_HLCTL - 0x001E

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HLCTL for PRT[0]	0x1E2000	
3:0	Type	0x2	



6.2.21.5.2 Starting Address High at PRTMAC_HLCTL - 0x001F

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HLCTL for PRT[0]		

6.2.21.5.3 Attributes at PRTMAC_HLCTL - 0x0020

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.6 PRTMAC_PAP - (0x0029 - 0033)

6.2.21.6.1 Starting Address Low at PRTMAC_PAP - 0x0029

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_PAP for PRT[0]	0x1E2040	
3:0	Type	0x2	

6.2.21.6.2 Starting Address High at PRTMAC_PAP - 0x002A

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_PAP for PRT[0]		

6.2.21.6.3 Attributes at PRTMAC_PAP - 0x002B

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.7 PRTGL_SAL - (0x0034 - 003E)

6.2.21.7.1 Starting Address Low at PRTGL_SAL - 0x0034

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTGL_SAL for PRT[0]	0x1E2120	



Bits	Field Name	Default NVM Value	Description
3:0	Type	0x2	

6.2.21.7.2 Starting Address High at PRTGL_SAL - 0x0035

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTGL_SAL for PRT[0]		

6.2.21.7.3 Attributes at PRTGL_SAL - 0x0036

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.7.4 Data Low of PRTGL_SAL[PRT] - 0x0037 + 2*PRT, PRT=0...3

6.2.21.7.5 Data High of PRTGL_SAL[PRT] - 0x0038 + 2*PRT, PRT=0...3

6.2.21.8 PRTGL_SAH - (0x003F - 0049)

6.2.21.8.1 Starting Address Low at PRTGL_SAH - 0x003F

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTGL_SAH for PRT[0]	0x1E2140	
3:0	Type	0x2	

6.2.21.8.2 Starting Address High at PRTGL_SAH - 0x0040

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTGL_SAH for PRT[0]		

6.2.21.8.3 Attributes at PRTGL_SAH - 0x0041

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.21.8.4 Data Low of PRTGL_SAH[PRT] - $0x0042 + 2 * PRT$, PRT=0...3

6.2.21.8.5 Data High of PRTGL_SAH[PRT] - $0x0043 + 2 * PRT$, PRT=0...3

6.2.21.9 PRTDCB_MFLCN - (0x004A - 0054)

6.2.21.9.1 Starting Address Low at PRTDCB_MFLCN - 0x004A

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_MFLCN for PRT[0]	0x1E2400	
3:0	Type	0x2	

6.2.21.9.2 Starting Address High at PRTDCB_MFLCN - 0x004B

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_MFLCN for PRT[0]		

6.2.21.9.3 Attributes at PRTDCB_MFLCN - 0x004C

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.9.4 Data Low of PRTDCB_MFLCN[PRT] - $0x004D + 2 * PRT$, PRT=0...3

6.2.21.9.5 Data High of PRTDCB_MFLCN[PRT] - $0x004E + 2 * PRT$, PRT=0...3

6.2.21.10 PRTMAC_MACC - (0x0055 - 005F)

6.2.21.10.1 Starting Address Low at PRTMAC_MACC - 0x0055

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_MACC for PRT[0]	0x1E24E0	
3:0	Type	0x2	



6.2.21.10.2 Starting Address High at PRTMAC_MACC - 0x0056

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_MACC for PRT[0]		

6.2.21.10.3 Attributes at PRTMAC_MACC - 0x0057

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.11 PRTMAC_HSEC_CTL_RX_MAX_PACKET_LEN - (0x0060 - 0066)

6.2.21.11.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_MAX_PACKET_LEN - 0x0060

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_MAX_PACKET_LEN for PRT2[0]	0x1E30A0	
3:0	Type	0x2	

6.2.21.11.2 Starting Address High at PRTMAC_HSEC_CTL_RX_MAX_PACKET_LEN - 0x0061 + 2*PRT2, PRT2=0

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_MAX_PACKET_LEN for PRT2[0]		

6.2.21.11.3 Attributes at PRTMAC_HSEC_CTL_RX_MAX_PACKET_LEN - 0x0062

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.21.12 PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE - (0x0067 - 006D)

6.2.21.12.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE - 0x0067

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE for PRT2[0]	0x1E30C0	
3:0	Type	0x2	

6.2.21.12.2 Starting Address High at PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE - 0x0068

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE for PRT2[0]		

6.2.21.12.3 Attributes at PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE - 0x0069

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

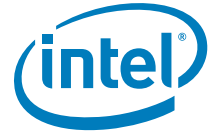
6.2.21.12.4 Data Low of PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE[PRT2] - 0x006A + 2*PRT2, PRT2=0...1

6.2.21.12.5 Data High of PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE[PRT2] - 0x006B + 2*PRT2, PRT2=0...1

6.2.21.13 PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE - (0x006E - 0074)

6.2.21.13.1 Starting Address Low at PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE - 0x006E

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE for PRT2[0]	0x1E30D0	
3:0	Type	0x2	



6.2.21.13.2 Starting Address High at PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE - 0x006F

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE for PRT2[0]		



6.2.21.13.3 Attributes at PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE - 0x0070

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.13.4 Data Low of PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE[PRT2] - 0x0071 + 2*PRT2, PRT2=0...1

6.2.21.13.5 Data High of PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE[PRT2] - 0x0072 + 2*PRT2, PRT2=0...1

6.2.21.14 PRTMAC_HSEC_CTL_RX_ENABLE_GCP - (0x0075 - 007B)

6.2.21.14.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_ENABLE_GCP - 0x0075

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_ENABLE_GCP for PRT2[0]	0x1E30E0	
3:0	Type	0x2	

6.2.21.14.2 Starting Address High at PRTMAC_HSEC_CTL_RX_ENABLE_GCP - 0x0076

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_ENABLE_GCP for PRT2[0]		

6.2.21.14.3 Attributes at PRTMAC_HSEC_CTL_RX_ENABLE_GCP - 0x0077

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.14.4 Data Low of PRTMAC_HSEC_CTL_RX_ENABLE_GCP[PRT2] - 0x0078 + 2*PRT2, PRT2=0...1

6.2.21.14.5 Data High of PRTMAC_HSEC_CTL_RX_ENABLE_GCP[PRT2] - 0x0079 + 2*PRT2, PRT2=0...1



6.2.21.15 PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GCP - (0x007C - 0082)

6.2.21.15.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GCP - 0x007C

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GCP for PRT2[0]	0x1E3100	
3:0	Type	0x2	

6.2.21.15.2 Starting Address High at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GCP - 0x007D

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GCP for PRT2[0]		

6.2.21.15.3 Attributes at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GCP - 0x007E

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.16 PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 - (0x0083 - 0089)

6.2.21.16.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 - 0x0083

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 for PRT2[0]	0x1E3110	
3:0	Type	0x2	

6.2.21.16.2 Starting Address High at PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 - 0x0084

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 for PRT2[0]		



6.2.21.16.3 Attributes at PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 - 0x0085

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.16.4 Data Low of PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1[PRT2] - 0x0086 + 2*PRT2, PRT2=0...1

6.2.21.16.5 Data High of PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1[PRT2] - 0x0087 + 2*PRT2, PRT2=0...1

6.2.21.17 PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 - (0x008A - 0090)

6.2.21.17.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 - 0x008A

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 for PRT2[0]	0x1E3120	
3:0	Type	0x2	

6.2.21.17.2 Starting Address High at PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 - 0x008B

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 for PRT2[0]		

6.2.21.17.3 Attributes at PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 - 0x008C

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.17.4 Data Low of PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2[PRT2] - 0x008D + 2*PRT2, PRT2=0...1

6.2.21.17.5 Data High of PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2[PRT2] - 0x008E + 2*PRT2, PRT2=0...1



6.2.21.18 PRTMAC_HSEC_CTL_RX_CHECK_SA_GCP - (0x0091 - 0097)

6.2.21.18.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_CHECK_SA_GCP - 0x0091

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_CHECK_SA_GCP for PRT2[0]	0x1E3130	
3:0	Type	0x2	

6.2.21.18.2 Starting Address High at PRTMAC_HSEC_CTL_RX_CHECK_SA_GCP - 0x0092

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_CHECK_SA_GCP for PRT2[0]		

6.2.21.18.3 Attributes at PRTMAC_HSEC_CTL_RX_CHECK_SA_GCP - 0x0093

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.19 PRTMAC_HSEC_CTL_RX_ENABLE_GPP - (0x0098 - 009E)

6.2.21.19.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_ENABLE_GPP - 0x0098

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_ENABLE_GPP for PRT2[0]	0x1E3260	
3:0	Type	0x2	

6.2.21.19.2 Starting Address High at PRTMAC_HSEC_CTL_RX_ENABLE_GPP - 0x0099

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_ENABLE_GPP for PRT2[0]		



6.2.21.19.3 Attributes at PRTMAC_HSEC_CTL_RX_ENABLE_GPP - 0x009A

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.19.4 Data Low of PRTMAC_HSEC_CTL_RX_ENABLE_GPP[PRT2] - 0x009B + 2*PRT2, PRT2=0...1

6.2.21.19.5 Data High of PRTMAC_HSEC_CTL_RX_ENABLE_GPP[PRT2] - 0x009C + 2*PRT2, PRT2=0...1

6.2.21.20 PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GPP - (0x009F - 00A5)

6.2.21.20.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GPP - 0x009F

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GPP for PRT2[0]	0x1E3280	
3:0	Type	0x2	

6.2.21.20.2 Starting Address High at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GPP - 0x00A0

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GPP for PRT2[0]		

6.2.21.20.3 Attributes at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_GPP - 0x00A1

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.21 PRTMAC_HSEC_CTL_RX_ENABLE_PPP - (0x00A6 - 00AC)

6.2.21.21.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_ENABLE_PPP - 0x00A6

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_ENABLE_PPP for PRT2[0]	0x1E32E0	



Bits	Field Name	Default NVM Value	Description
3:0	Type	0x2	

6.2.21.21.2 Starting Address High at PRTMAC_HSEC_CTL_RX_ENABLE_PPP - 0x00A7

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_ENABLE_PPP for PRT2[0]		



6.2.21.21.3 Attributes at PRTMAC_HSEC_CTL_RX_ENABLE_PPP - 0x00A8

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.21.4 Data Low of PRTMAC_HSEC_CTL_RX_ENABLE_PPP[PRT2] - 0x00A9 + 2*PRT2, PRT2=0...1

6.2.21.21.5 Data High of PRTMAC_HSEC_CTL_RX_ENABLE_PPP[PRT2] - 0x00AA + 2*PRT2, PRT2=0...1

6.2.21.22 PRTMAC_HSEC_CTL_RX_CHECK_UCAST_PPP - (0x00AD - 00B3)

6.2.21.22.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_PPP - 0x00AD

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_CHECK_UCAST_PPP for PRT2[0]	0x1E3300	
3:0	Type	0x2	

6.2.21.22.2 Starting Address High at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_PPP - 0x00AE

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_CHECK_UCAST_PPP for PRT2[0]		

6.2.21.22.3 Attributes at PRTMAC_HSEC_CTL_RX_CHECK_UCAST_PPP - 0x00AF

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.21.23 PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL - (0x00B4 - 00BA)

6.2.21.23.1 Starting Address Low at PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL - 0x00B4

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL for PRT2[0]	0x1E3360	
3:0	Type	0x2	

6.2.21.23.2 Starting Address High at PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL - 0x00B5

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL for PRT2[0]		

6.2.21.23.3 Attributes at PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL - 0x00B6

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.23.4 Data Low of PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL[PRT2] - 0x00B7 + 2*PRT2, PRT2=0...1

6.2.21.23.5 Data High of PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL[PRT2] - 0x00B8 + 2*PRT2, PRT2=0...1

6.2.21.24 PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA - (0x00BB - 00F9)

6.2.21.24.1 Starting Address Low at PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA - 0x00BB

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA for PRT2[0]	0x1E3370	
3:0	Type	0x2	



6.2.21.24.2 Starting Address High at PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA - 0x00BC

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA for PRT2[0]		

6.2.21.24.3 Attributes at PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA - 0x00BD

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.24.4 Data Low of PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA[n,PRT2] - 0x00BE + 7*n + 2*PRT2, n=0...8, PRT2=0...1

6.2.21.24.5 Data High of PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA[n,PRT2] - 0x00BF + 7*n + 2*PRT2, n=0...8, PRT2=0...1

6.2.21.25 PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER - (0x00FA - 0138)

6.2.21.25.1 Starting Address Low at PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER - 0x00FA

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER for PRT2[0]	0x1E3400	
3:0	Type	0x2	

6.2.21.25.2 Starting Address High at PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER - 0x00FB

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER for PRT2[0]		

6.2.21.25.3 Attributes at PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER - 0x00FC

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.21.25.4 Data Low of PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER[n,PRT2] - 0x00FD + 7*n + 2*PRT2, n=0...8, PRT2=0...1

6.2.21.25.5 Data High of PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER[n,PRT2] - 0x00FE + 7*n + 2*PRT2, n=0...8, PRT2=0...1

6.2.21.26 PRTMAC_HSEC_CTL_TX_SA_PART1 - (0x0139 - 013F)

6.2.21.26.1 Starting Address Low at PRTMAC_HSEC_CTL_TX_SA_PART1 - 0x0139

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_TX_SA_PART1 for PRT2[0]	0x1E34B0	
3:0	Type	0x2	

6.2.21.26.2 Starting Address High at PRTMAC_HSEC_CTL_TX_SA_PART1 - 0x013A

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_TX_SA_PART1 for PRT2[0]		

6.2.21.26.3 Attributes at PRTMAC_HSEC_CTL_TX_SA_PART1 - 0x013B

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.26.4 Data Low of PRTMAC_HSEC_CTL_TX_SA_PART1[PRT2] - 0x013C + 2*PRT2, PRT2=0...1

6.2.21.26.5 Data High of PRTMAC_HSEC_CTL_TX_SA_PART1[PRT2] - 0x013D + 2*PRT2, PRT2=0...1

6.2.21.27 PRTMAC_HSEC_CTL_TX_SA_PART2 - (0x0140 - 0146)

6.2.21.27.1 Starting Address Low at PRTMAC_HSEC_CTL_TX_SA_PART2 - 0x0140

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_TX_SA_PART2 for PRT2[0]	0x1E34C0	
3:0	Type	0x2	



6.2.21.27.2 Starting Address High at PRTMAC_HSEC_CTL_TX_SA_PART2 - 0x0141

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_TX_SA_PART2 for PRT2[0]		

6.2.21.27.3 Attributes at PRTMAC_HSEC_CTL_TX_SA_PART2 - 0x0142

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.27.4 Data Low of PRTMAC_HSEC_CTL_TX_SA_PART2[PRT2] - 0x0143 + 2*PRT2, PRT2=0...1

6.2.21.27.5 Data High of PRTMAC_HSEC_CTL_TX_SA_PART2[PRT2] - 0x0144 + 2*PRT2, PRT2=0...1

6.2.21.28 PRTMAC_HSEC_CTL_INTERNAL - (0x0147 - 014D)

6.2.21.28.1 Starting Address Low at PRTMAC_HSEC_CTL_INTERNAL - 0x0147

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_INTERNAL for PRT2[0]	0x1E3530	
3:0	Type	0x2	

6.2.21.28.2 Starting Address High at PRTMAC_HSEC_CTL_INTERNAL - 0x0148

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_INTERNAL for PRT2[0]		

6.2.21.28.3 Attributes at PRTMAC_HSEC_CTL_INTERNAL - 0x0149

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.21.29 PRTMAC_HSEC_CTL_XLGMII - (0x0152 - 0158)

6.2.21.29.1 Starting Address Low at PRTMAC_HSEC_CTL_XLGMII - 0x0152 + 2*PRT2, PRT2=0

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSEC_CTL_XLGMII for PRT2[0]	0x1E3550	
3:0	Type	0x2	

6.2.21.29.2 Starting Address High at PRTMAC_HSEC_CTL_XLGMII - 0x0153

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSEC_CTL_XLGMII for PRT2[0]		

6.2.21.29.3 Attributes at PRTMAC_HSEC_CTL_XLGMII - 0x0154

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.30 PRTMAC_HSECTL1 - (0x0159 - 015F)

6.2.21.30.1 Starting Address Low at PRTMAC_HSECTL1 - 0x0159

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HSECTL1 for PRT2[0]	0x1E3560	
3:0	Type	0x2	

6.2.21.30.2 Starting Address High at PRTMAC_HSECTL1 - 0x015A

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HSECTL1 for PRT2[0]		

6.2.21.30.3 Attributes at PRTMAC_HSECTL1 - 0x015B

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	



Bits	Field Name	Default NVM Value	Description
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.31 PRTTSYN_CTL0 - (0x0160 - 016A)

6.2.21.31.1 Starting Address Low at PRTTSYN_CTL0 - 0x0160

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTTSYN_CTL0 for PRT[0]	0x1E4200	
3:0	Type	0x2	

6.2.21.31.2 Starting Address High at PRTTSYN_CTL0 - 0x0161

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTTSYN_CTL0 for PRT[0]		

6.2.21.31.3 Attributes at PRTTSYN_CTL0 - 0x0162

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.31.4 Data Low of PRTTSYN_CTL0[PRT] - 0x0163 + 2*PRT, PRT=0...3

6.2.21.31.5 Data High of PRTTSYN_CTL0[PRT] - 0x0164 + 2*PRT, PRT=0...3



6.2.21.32 GLPM_EEE_SU - (0x016B - 016F)

6.2.21.32.1 Starting Address Low at GLPM_EEE_SU - 0x016B

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of GLPM_EEE_SU	0x1E4340	
3:0	Type	0x2	

6.2.21.32.2 Starting Address High at GLPM_EEE_SU - 0x016C

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of GLPM_EEE_SU		

6.2.21.32.3 Attributes at GLPM_EEE_SU - 0x016D

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x2	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.33 PRTPM_EEER - (0x0172 - 017C)

6.2.21.33.1 Starting Address Low at PRTPM_EEER - 0x0172

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTPM_EEER for PRT[0]	0x1E4360	
3:0	Type	0x2	

6.2.21.33.2 Starting Address High at PRTPM_EEER - 0x0173

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTPM_EEER for PRT[0]		

6.2.21.33.3 Attributes at PRTPM_EEER - 0x0174

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	



6.2.21.33.4 Data Low of PRTPM_EEER[PRT] - $0x0175 + 2 * PRT$, PRT=0...3

6.2.21.33.5 Data High of PRTPM_EEER[PRT] - $0x0176 + 2 * PRT$, PRT=0...3

6.2.21.34 PRTPM_EEEC - (0x017D - 0187)

6.2.21.34.1 Starting Address Low at PRTPM_EEEC - 0x017D

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTPM_EEEC for PRT[0]	0x1E4380	
3:0	Type	0x2	

6.2.21.34.2 Starting Address High at PRTPM_EEEC - 0x017E

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTPM_EEEC for PRT[0]		

6.2.21.34.3 Attributes at PRTPM_EEEC - 0x017F

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.34.4 Data Low of PRTPM_EEEC[PRT] - $0x0180 + 2 * PRT$, PRT=0...3

6.2.21.34.5 Data High of PRTPM_EEEC[PRT] - $0x0181 + 2 * PRT$, PRT=0...3

6.2.21.35 PRTDCB_FCTTVN - (0x0188 - 01B3)

6.2.21.35.1 Starting Address Low at PRTDCB_FCTTVN - 0x0188

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_FCTTVN for PRT[0]	0x1E4580	
3:0	Type	0x2	



6.2.21.35.2 Starting Address High at PRTDCB_FCTTVN - 0x0189

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_FCTTVN for PRT[0]		

6.2.21.35.3 Attributes at PRTDCB_FCTTVN - 0x018A

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.35.4 Data Low of PRTDCB_FCTTVN[n,PRT] - 0x018B + 11*n + 2*PRT, n=0...3, PRT=0...3

6.2.21.35.5 Data High of PRTDCB_FCTTVN[n,PRT] - 0x018C + 11*n + 2*PRT, n=0...3, PRT=0...3

6.2.21.36 PRTDCB_FCRTV - (0x01B4 - 01BE)

6.2.21.36.1 Starting Address Low at PRTDCB_FCRTV - 0x01B4

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_FCRTV for PRT[0]	0x1E4600	
3:0	Type	0x2	

6.2.21.36.2 Starting Address High at PRTDCB_FCRTV - 0x01B5

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_FCRTV for PRT[0]		

6.2.21.36.3 Attributes at PRTDCB_FCRTV - 0x01B6

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.36.4 Data Low of PRTDCB_FCRTV[PRT] - 0x01B7 + 2*PRT, PRT=0...3

6.2.21.36.5 Data High of PRTDCB_FCRTV[PRT] - 0x01B8 + 2*PRT, PRT=0...3



6.2.21.37 PRTDCB_FCCFG - (0x01BF - 01C9)

6.2.21.37.1 Starting Address Low at PRTDCB_FCCFG - 0x01BF

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTDCB_FCCFG for PRT[0]	0x1E4640	
3:0	Type	0x2	

6.2.21.37.2 Starting Address High at PRTDCB_FCCFG - 0x01C0

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTDCB_FCCFG for PRT[0]		

6.2.21.37.3 Attributes at PRTDCB_FCCFG - 0x01C1

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.21.37.4 Data Low of PRTDCB_FCCFG[PRT] - 0x01C2 + 2*PRT, PRT=0...3

6.2.21.37.5 Data High of PRTDCB_FCCFG[PRT] - 0x01C3 + 2*PRT, PRT=0...3

6.2.21.38 PRTMAC_HLCTLA - (0x01CA - 01D4)

6.2.21.38.1 Starting Address Low at PRTMAC_HLCTLA - 0x01CA

Bits	Field Name	Default NVM Value	Description
15:4	Low Address Bits of PRTMAC_HLCTLA for PRT[0]	0x1E4760	
3:0	Type	0x2	

6.2.21.38.2 Starting Address High at PRTMAC_HLCTLA - 0x01CB

Bits	Field Name	Default NVM Value	Description
15:0	High Address Bits of PRTMAC_HLCTLA for PRT[0]		



6.2.21.38.3 Attributes at PRTMAC_HLCTLA - 0x01CC

Bits	Field Name	Default NVM Value	Description
15:5	Length	0x4	
4:3	Skip	0x0	
2:0	Width	0x0	

6.2.22 EMP SR Settings Module Header Section Summary Table

This section contains the modes of operation of the EMP which shall be stored in the shadow RAM.

Word Offset	Description	Reference
0x0000	Section Header	See section 6.2.22.1
0x0001	SMBus Slave Addresses	See section 6.2.22.2
0x0002	SMBus Slave Addresses 2	See section 6.2.22.3
0x0005	SR PF Allocations Pointer	See section 6.2.22.4
0x0006	Max PF and VF Per Port	See section 6.2.22.5
0x0007	PF LAN Device ID	See section 6.2.22.6
0x0008	PF SAN Device ID	See section 6.2.22.7
0x0009	PF iSCSI Device ID	See section 6.2.22.8
0x000B	VF LAN Device ID	See section 6.2.22.9
0x000C	VF SAN Device ID	See section 6.2.22.10
0x000D	VF iSCSI Device ID	See section 6.2.22.11
0x000F	PF LAN Subsystem ID	See section 6.2.22.12
0x0010	PF SAN Subsystem ID	See section 6.2.22.13
0x0011	PF iSCSI Subsystem ID	See section 6.2.22.14
0x0013	VF LAN Subsystem ID	See section 6.2.22.15
0x0014	VF SAN Subsystem ID	See section 6.2.22.16
0x0015	VF iSCSI Subsystem ID	See section 6.2.22.17
0x0017	MNG MAC Address Pointer	See section 6.2.22.18
0x0018	PF MAC Address Pointer	See section 6.2.22.19
0x0019	PHY Capability LAN 0 Pointer	See section 6.2.22.20
0x001A	PHY Capability LAN 1 Pointer	See section 6.2.22.21
0x001B	PHY Capability LAN 2 Pointer	See section 6.2.22.22
0x001C	PHY Capability LAN 3 Pointer	See section 6.2.22.23
0x001D + 1*n, n=0...7	PFGEN_STATE	See section 6.2.22.24
0x0025	EXT to INT PHY Mapping LAN 0 Pointer	See section 6.2.22.25
0x0026	EXT to INT PHY Mapping LAN 1 Pointer	See section 6.2.22.26
0x0027	EXT to INT PHY Mapping LAN 2 Pointer	See section 6.2.22.27
0x0028	EXT to INT PHY Mapping LAN 3 Pointer	See section 6.2.22.28



Word Offset	Description	Reference
0x0029	OEM Current Settings Pointer	See section 6.2.22.29
0x002A	Features Enable	See section 6.2.22.29
0x002B	Current Settings Pointer	

6.2.22.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Block Length		Length in: 2 bytes unit - 1 First Section -> Word: EMP SR Settings Module Header -> Section Header Last Section -> Word: EMP SR Settings Module Header -> Current Settings Pointer

6.2.22.2 SMBus Slave Addresses - 0x0001

Bits	Field Name	Default NVM Value	Description
15:9	SMBus 1 Slave Address	0x4A	Dual address mode only. Note: This field is preserved by the Intel NVM update tool.
8	Reserved	0x0	Reserved.
7:1	SMBus 0 Slave Address	0x49	Note: This field is preserved by the Intel NVM update tool.
0	Reserved	0x0	Reserved.

6.2.22.3 SMBus Slave Addresses 2 - 0x0002

Bits	Field Name	Default NVM Value	Description
15:9	SMBus 3 Slave Addresses	0x4C	Note: This field is preserved by the Intel NVM update tool.
8	res	0x0	Reserved.
7:1	SMBus 2 Slave Addresses	0x4B	Note: This field is preserved by the Intel NVM update tool.
0	Reserved	0x0	Reserved.

6.2.22.4 SR PF Allocations Pointer - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	PF Allocations Pointer	0xFFFF	

The PF allocations pointer points to the SR PF Allocations section. For SR PF Allocations inner structure detail, see [Section 6.2.23](#).



6.2.22.5 Max PF and VF per Port - 0x0006

Bits	Field Name	Default NVM Value	Description
15:8	Max VF per PF	0x80	Maximum number of VFs allocated to a PF. Note: This field is preserved by the Intel NVM update tool.
7:3	Reserved	0x00	Reserved.
2:0	Max PF per Port	0x2	Maximum PF per Port. 0x0 = 1 PF per port. 0x1 = 2 PFs per port. 0x2 = 4 PFs per port. 0x3 = 8 PFs per port. 0x4 = 16 PFs per port. 0x5 = Reserved. 0x6 = Reserved. 0x7 = Reserved. Note: This field is preserved by the Intel NVM update tool.

6.2.22.6 PF LAN Device ID - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	PF LAN Device ID	0x154B	PF LAN Device ID. 0x154B = Default silicon default. 0x1572 = SFI SFI. 0x1580 = Reserved. 0x1581 = 10 GbE backplane. 0x1583 = Reserved. 0x1584 = Reserved. 0x1586 = 10GBASE-T. 0x1587 = Reserved. 0x1588 = Reserved. 0x1589 = Reserved. 0x158A = Reserved. 0x158B = Reserved. 0x15FF = X710-TM4 10GBASE-T. 0x104E = X710-TM4 10 GbE SFP+. 0x104F = X710-TM4 10 GbE backplane. Note: This field is preserved by the Intel NVM update tool.

6.2.22.7 PF SAN Device ID - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	PF SAN Device ID	0x154B	Note: This field is preserved by the Intel NVM update tool.

6.2.22.8 PF iSCSI Device ID - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	PF iSCSI Device ID	0x154B	Note: This field is preserved by the Intel NVM update tool.



6.2.22.9 VF LAN Device ID - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	VF LAN Device ID	0x154C	Note: This field is preserved by the Intel NVM update tool.

6.2.22.10 VF SAN Device ID - 0x000C

Bits	Field Name	Default NVM Value	Description
15:0	VF SAN Device ID	0x154C	Note: This field is preserved by the Intel NVM update tool.

6.2.22.11 VF iSCSI Device ID - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	VF iSCSI Device ID	0x154C	Note: This field is preserved by the Intel NVM update tool.

6.2.22.12 PF LAN Subsystem ID - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	PF LAN Subsystem ID	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.22.13 PF SAN Subsystem ID - 0x0010

Bits	Field Name	Default NVM Value	Description
15:0	PF SAN Subsystem ID	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.22.14 PF iSCSI Subsystem ID - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	PF iSCSI Subsystem ID	0x154B	Note: This field is preserved by the Intel NVM update tool.

6.2.22.15 VF LAN Subsystem ID - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	VF LAN Subsystem ID	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.22.16 VF SAN Subsystem ID - 0x0014

Bits	Field Name	Default NVM Value	Description
15:0	VF SAN Subsystem ID	0x0	Note: This field is preserved by the Intel NVM update tool.



6.2.22.17 VF iSCSI Subsystem ID - 0x0015

Bits	Field Name	Default NVM Value	Description
15:0	VF iSCSI Subsystem ID	0x154B	Note: This field is preserved by the Intel NVM update tool.

6.2.22.18 MNG MAC Address Pointer - 0x0017

Bits	Field Name	Default NVM Value	Description
15:0	MNG MAC Address Pointer	0xFFFF	

The MNG MAC Address pointer points to the MNG MAC Address section. For MNG MAC Address inner structure, see [Section 6.2.24](#).

6.2.22.19 PF MAC Address Pointer - 0x0018

Bits	Field Name	Default NVM Value	Description
15:0	PF MAC Address pointer	0xFFFF	

The PF MAC Address pointer points to the PF MAC Address section. For PF MAC Address inner structure detail, see [Section 6.2.25](#).

6.2.22.20 PHY Capability LAN 0 Pointer - 0x0019

Bits	Field Name	Default NVM Value	Description
15:0	PHY	0xFFFF	

The PHY points to the PHY Capability Data Structure 0 Section. For PHY Capability Data Structure 0 inner structure detail, see [Section 6.2.26](#).

6.2.22.21 PHY Capability LAN 1 Pointer - 0x001A

Bits	Field Name	Default NVM Value	Description
15:0	PHY	0xFFFF	

The PHY points to PHY Capability Data Structure 1 section. For PHY Capability Data Structure 1 inner structure detail, see [Section 6.2.26](#).

6.2.22.22 PHY Capability LAN 2 Pointer - 0x001B

Bits	Field Name	Default NVM Value	Description
15:0	PHY	0xFFFF	

The PHY points to the PHY Capability Data Structure 2 section. For PHY Capability Data Structure 2 inner structure detail, see [Section 6.2.26](#).



6.2.22.23 PHY Capability LAN 3 Pointer - 0x001C

Bits	Field Name	Default NVM Value	Description
15:0	PHY	0xFFFF	

The PHY points to the PHY Capability Data Structure 3 section. For PHY Capability Data Structure 3 inner structure detail, see [Section 6.2.26](#).

6.2.22.24 PFGEN_STATE[n] (0x001D + 1*n, n=0...7)

Bits	Field Name	Default NVM Value	Description
15:12	Reserved	0x0	Reserved.
11	PFGEN_STATE[1 + 2*n].PFSCEN	0x0	Note: This field is preserved by the Intel NVM update tool.
10	PFGEN_STATE[1 + 2*n].PFLINKEN	0x1	Note: This field is preserved by the Intel NVM update tool.
9	PFGEN_STATE[1 + 2*n].PFFCEN	0x1	Note: This field is preserved by the Intel NVM update tool.
7:4	Reserved	0x0	Reserved.
3	PFGEN_STATE[0 + 2*n].PFSCEN	0x0	Note: This field is preserved by the Intel NVM update tool.
2	PFGEN_STATE[0 + 2*n].PFLINKEN	0x1	Note: This field is preserved by the Intel NVM update tool.
1	PFGEN_STATE[0 + 2*n].PFFCEN	0x1	Note: This field is preserved by the Intel NVM update tool.

6.2.22.25 EXT to INT PHY Mapping LAN 0 Pointer - 0x0025

Bits	Field Name	Default NVM Value	Description
15:0	LAN 0 Pointer	0xFFFF	

The LAN 0 Pointer points to the External to Internal PHY Mapping 0 section. For External to Internal PHY Mapping 0 inner structure detail, see [Section 6.2.29](#).

6.2.22.26 EXT to INT PHY Mapping LAN 1 Pointer - 0x0026

Bits	Field Name	Default NVM Value	Description
15:0	LAN 0 Pointer	0xFFFF	

The LAN 0 Pointer points to the External to Internal PHY Mapping 1 section. For External to Internal PHY Mapping 1 inner structure detail, see [Section 6.2.29](#).



6.2.22.27 EXT to INT PHY Mapping LAN 2 Pointer - 0x0027

Bits	Field Name	Default NVM Value	Description
15:0	LAN 0 Pointer	0xFFFF	

The LAN 0 Pointer points to the External to Internal PHY Mapping 2 section. For External to Internal PHY Mapping 2 inner structure detail, see [Section 6.2.29](#).

6.2.22.28 EXT to INT PHY Mapping LAN 3 Pointer - 0x0028

Bits	Field Name	Default NVM Value	Description
15:0	LAN 0 Pointer	0xFFFF	

The LAN 0 Pointer points to the External to Internal PHY Mapping 3 section. For External to Internal PHY Mapping 3 inner structure detail, see [Section 6.2.29](#).

6.2.22.29 Features Enable - 0x002A

Describes support for various features.

Bits	Field Name	Default NVM Value	Description
15:13	Reserved	0x0	Reserved.
12	Channel Identifier	0x0	Defines the Ethertype of the channel identifier used to differentiate channels within a port extender. 0x0 = S-tag. 0x1 = VLAN. Note: This field is preserved by the Intel NVM update tool.
11:9	Switching Mode	0x0	Reserved. 0x0 = EVB switching 802.1Qbg switching.
8	Proxy Enabled for Port 3	0x1	If cleared, proxy functionality embedded in the EMP is disabled on Port 3. It could be re-enabled on this port only by resetting this bit to 1b and performing an EMPR cycle. 0x0 = Disabled. 0x1 = Enabled. Note: This field is preserved by the Intel NVM update tool.
7	Proxy Enabled for Port 2	0x1	If cleared, proxy functionality embedded in the EMP is disabled on Port 2. It could be re-enabled on this port only by resetting this bit to 1b and performing an EMPR cycle. 0x0 = Disabled. 0x1 = Enabled. Note: This field is preserved by the Intel NVM update tool.
6	Proxy Enabled for Port 1	0x1	If cleared, proxy functionality embedded in the EMP is disabled on Port 1. It could be re-enabled on this port only by resetting this bit to 1b and performing an EMPR cycle. 0x0 = Disabled. 0x1 = Enabled. Note: This field is preserved by the Intel NVM update tool.
5	Proxy Enabled for Port 0	0x1	If cleared, proxy functionality embedded in the EMP is disabled on Port 0. It could be re-enabled on this port only by resetting this bit to 1b and performing an EMPR cycle. 0x0 = Disabled. 0x1 = Enabled. Note: This field is preserved by the Intel NVM update tool.
4:3	Reserved	0x0	Reserved.



Bits	Field Name	Default NVM Value	Description
1	EVB Protocols Enabled	0x1	If set, filtering according to the IEEE 802.1QBg specification is enabled. 0x0 = Disabled. 0x1 = Enabled. Note: This field is preserved by the Intel NVM update tool.
0	VEB Statistics Disable	0x1	When clear (default), VEB statistics are enabled. When set, VEB statistics are disabled. Note: This field is preserved by the Intel NVM update tool.

6.2.22.30 Current Settings Pointer - 0x002B

Bits	Field Name	Default NVM Value	Description
15:0	Current Settings Pointer	0xFFFF	

The Current Settings Pointer points to the Current Settings section. For Current Settings Pointer inner structure detail, see [Section 6.2.29](#).

6.2.23 SR PF Allocations Section Summary Table

This section contains the allocation of resources per PF that must be stored in the shadow RAM.

Word Offset	Description	Reference
0x0000	Section Header	See section 6.2.23.1
0x0001 + 11*n, n=0...15	PF Flags	See section 6.2.23.2
0x0002 + 11*n, n=0...15	PF BW	See section 6.2.23.3
0x0003 + 11*n, n=0...15	PF Allocations - VSI and VEB	See section 6.2.23.4
0x0004 + 11*n, n=0...15	PF Allocations - MACs	See section 6.2.23.5
0x0005 + 11*n, n=0...15	PF Allocations - Outer Tags	See section 6.2.23.6
0x0006 + 11*n, n=0...15	Reserved	See section 6.2.23.7
0x0007 + 11*n, n=0...15	PF Allocations - Inner MACs	See section 6.2.23.8
0x0008 + 11*n, n=0...15	PF Allocations - IPs	See section 6.2.23.9
0x0009 + 11*n, n=0...15	PF Allocations - Stats	See section 6.2.23.10
0x000A + 11*n, n=0...15	PF Allocations - Mirror Rules	See section 6.2.23.11
0x000B + 11*n, n=0...15	PF Allocations - Queue Sets	See section 6.2.23.12



6.2.23.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Block Length		Length in: 2 bytes unit - 1 First Section -> Word: SR PF Allocations -> Section Header Last Section -> Word: SR PF Allocations -> PF Allocations - Queue Sets Section Length in Words

6.2.23.2 PF Flags[n] - 0x0001 + 11*n, n=0...15

Bits	Field Name	Default NVM Value	Description
15:3	Reserved	0x0	Reserved.
2	Multi_Qset	0x0	Note: This field is preserved by the Intel NVM update tool.
1	Reserved	0x0	Reserved.
0	Load PF MAC Address	0x1	Defines if the LAN MAC address of the PF should be added to the filtering table. If this bit is set, only packets that passes the MAC and if needed, the S-tag are forwarded to the PF. Note: This field is preserved by the Intel NVM update tool.

6.2.23.3 PF BW[n] - 0x0002 + 11*n, n=0...15

Bits	Field Name	Default NVM Value	Description
15:8	PF Max BW	0x64	This field contains the maximum Tx bandwidth allocation of the specified partition expressed in percent of the maximum physical port link speed. The percent value ranges from 0 to 100. Note: This field is preserved by the Intel NVM update tool.
7:0	PF Min BW	0x0	This field contains the minimum Tx bandwidth allocation of the specified partition expressed in percent of the maximum physical port link speed. The percent value ranges from 0 to 100. Note: This field is preserved by the Intel NVM update tool.

6.2.23.4 PF Allocations - VSI and VEB[n] - 0x0003 + 11*n, n=0...15

Defines the allocation of VSIs and VEBs to each PF.

Bits	Field Name	Default NVM Value	Description
15:10	VEBs	0x0	Defines the number of VEBs guaranteed to this PF. A value of zero means no VEBs are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.
9:0	VSIs	0x0	Defines the number of VSIs guaranteed to this PF. A value of zero means no VSIs are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.

**6.2.23.5 PF Allocations - MACs[n] - 0x0004 + 11*n, n=0...15**

Defines the allocation of MACs to each PF.

Bits	Field Name	Default NVM Value	Description
15:11	Reserved	0x0	Reserved. Note: This field is preserved by the Intel NVM update tool.
10:0	MACs	0x0	Defines the number of MACs guaranteed to this PF. A value of zero means no MACs are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.

6.2.23.6 PF Allocations - Outer Tags[n] - 0x0005 + 11*n, n=0...15

Defines the allocation of S-tags/inner VLANs/outer VLANs to each PF. The type of resource allocated depends on the switching table configuration.

Bits	Field Name	Default NVM Value	Description
15:10	Reserved	0x0	Reserved.
9:0	S-tags/Outer VLANs	0x0	Defines the number of outer tags guaranteed to this PF. A value of zero means no tags are guaranteed to this PF. The tags allocated by this field depends on the image used: 1. For EVB image - indicates the number of S-tags allocated to the PF. 2. For cloud images - N/A 3. For CPM image - indicates the number of outer VLANs allocated to the PF. Note: This field is preserved by the Intel NVM update tool.

6.2.23.7 Reserved[n] - 0x0006 + 11*n, n=0...15

Bits	Field Name	Default NVM Value	Description
15:10	Reserved	0x0	Reserved.
9:0	Inner VLANs	0x0	Defines the number of inner VLAN tags guaranteed to this PF. A value of zero means no tags are guaranteed to this PF. The tags allocated by this field depends on the image used: 1. For EVB image - N/A 2. For cloud images - indicates the number of inner VLANs allocated to the PF. 3. For OVEB image - indicates the number of regular VLANs allocated to the PF.

6.2.23.8 PF Allocations - Inner MACs[n] - 0x0007 + 11*n, n=0...15

Defines the allocation of inner MACs to each PF - relevant only for cloud images.

Bits	Field Name	Default NVM Value	Description
15:11	Reserved	0x0	Reserved.
10:0	Inner MACs	0x0	Defines the number of inner MACs guaranteed to this PF. A value of zero means no inner MACs are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.



6.2.23.9 PF Allocations - IPs[n] - 0x0008 + 11*n, n=0...15

Defines the allocation of IP to each PF - relevant only for cloud images.

Bits	Field Name	Default NVM Value	Description
15:11	Reserved	0x0	Reserved.
10:0	IPs	0x0	Defines the number of IP addresses guaranteed to this PF. A value of zero means no IP addresses are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.

6.2.23.10 PF Allocations - Stats [n] - 0x0009 + 11*n, n=0...15

Defines the allocation of statistics pools to each PF

Bits	Field Name	Default NVM Value	Description
15:14	Reserved	0x0	Reserved.
13:7	smonPrioStats Pools	0x0	Defines the number of smonPrioStats statistics pools guaranteed to this PF. A value of zero means no pools are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.
6:0	smonVlanStats Pools	0x0	Defines the number of smonVlanStats statistics pools guaranteed to this PF. A value of zero means no pools are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.

6.2.23.11 PF Allocations - Mirror Rules[n] - 0x000A + 11*n, n=0...15

Defines the allocation of mirror rules to each PF.

Bits	Field Name	Default NVM Value	Description
15:7	Reserved	0x0	Reserved.
6:0	Mirror Rules	0x0	Defines the number of mirror rules pools guaranteed to this PF. A value of zero means no rules are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.

6.2.23.12 PF Allocations - Queue Sets[n] - 0x000B + 11*n, n=0...15

Defines the allocation of Tx queue sets to each PF.

Bits	Field Name	Default NVM Value	Description
15:12	Reserved	0x0	Reserved.
11:0	Queue Sets	0x0	Defines the number of queue sets guaranteed to this PF. A value of zero means no queue sets are guaranteed to this PF. Note: This field is preserved by the Intel NVM update tool.



6.2.24 MNG MAC Address Section Summary Table

Word Offset	Description	Reference
0x0000	Section Header	See section 6.2.24.1
0x0001 + 3*n, n=0...3	LAN Ethernet MAC Address (LSB) MMAL	See section 6.2.24.2
0x0002 + 3*n, n=0...3	LAN Ethernet MAC Address (Mid) MMAL	See section 6.2.24.3
0x0003 + 3*n, n=0...3	LAN Ethernet MAC Address (MSB) MMAH	See section 6.2.24.4
0x000D + 3*n, n=0...3	LAN Ethernet MAC Address (LSB) MMAL	See section 6.2.24.5
0x000E + 3*n, n=0...3	LAN Ethernet MAC Address (Mid) MMAL	See section 6.2.24.6
0x000F + 3*n, n=0...3	LAN Ethernet MAC Address (MSB) MMAH	See section 6.2.24.7
0x0019 + 3*n, n=0...3	LAN Ethernet MAC Address (LSB) MMAL	See section 6.2.24.8
0x001A + 3*n, n=0...3	LAN Ethernet MAC Address (Mid) MMAL	See section 6.2.24.9
0x001B + 3*n, n=0...3	LAN Ethernet MAC Address (MSB) MMAH	See section 6.2.24.10
0x0025 + 3*n, n=0...3	LAN Ethernet MAC Address (LSB) MMAL	See section 6.2.24.11
0x0026 + 3*n, n=0...3	LAN Ethernet MAC Address (Mid) MMAL	See section 6.2.24.12
0x0027 + 3*n, n=0...3	LAN Ethernet MAC Address (MSB) MMAH	See section 6.2.24.13

6.2.24.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Block Length		Length in: 2 bytes unit - 1 First Section -> Word: MNG MAC Address -> Section Header Last Section -> Word: MNG MAC Address -> LAN Ethernet MAC Address (MSB) MMAH Section Length in Words

6.2.24.2 LAN Ethernet MAC Address (LSB) MMAL[n] - 0x0001 + 3*n, n=0...3

This word is loaded by firmware to the 16 LS bits of the MMAL[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

Bits	Field Name	Default NVM Value	Description
15:8	Ethernet MAC Address, Byte 1	0xFF	Note: This field is preserved by the Intel NVM update tool.
7:0	Ethernet MAC Address, Byte 0	0xFF	Note: This field is preserved by the Intel NVM update tool.



6.2.24.3 LAN Ethernet MAC Address (Mid) MMAL[n] - 0x0002 + 3*n, n=0...3

This word is loaded by firmware to the 16 MS bits of the MMAL[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

Bits	Field Name	Default NVM Value	Description
15:8	Ethernet MAC Address, Byte 3	0xFF	Note: This field is preserved by the Intel NVM update tool.
7:0	Ethernet MAC Address, Byte 2	0xFF	Note: This field is preserved by the Intel NVM update tool.

6.2.24.4 LAN Ethernet MAC Address (MSB) MMAH[n] - 0x0003 + 3*n, n=0...3

This word is loaded by firmware to the MMAH[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

Bits	Field Name	Default NVM Value	Description
15:8	Ethernet MAC Address, Byte 5	0xFF	Note: This field is preserved by the Intel NVM update tool.
7:0	Ethernet MAC Address, Byte 4	0xFF	Note: This field is preserved by the Intel NVM update tool.

6.2.24.5 LAN Ethernet MAC Address (LSB) MMAL[n] - 0x000D + 3*n, n=0...3

This word is loaded by firmware to the 16 LS bits of the MMAL[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.2](#).

6.2.24.6 LAN Ethernet MAC Address (Mid) MMAL[n] - 0x000E + 3*n, n=0...3

This word is loaded by firmware to the 16 MS bits of the MMAL[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.3](#).



6.2.24.7 LAN Ethernet MAC Address (MSB) MMAH[n] - 0x000F + 3*n, n=0...3

This word is loaded by firmware to the MMAH[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.4](#).

6.2.24.8 LAN Ethernet MAC Address (LSB) MMAL[n] - 0x0019 + 3*n, n=0...3

This word is loaded by firmware to the 16 LS bits of the MMAL[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.2](#).

6.2.24.9 LAN Ethernet MAC Address (Mid) MMAL[n] - 0x001A + 3*n, n=0...3

This word is loaded by firmware to the 16 MS bits of the MMAL[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.3](#).

6.2.24.10 LAN Ethernet MAC Address (MSB) MMAH[n] - 0x001B + 3*n, n=0...3

This word is loaded by firmware to the MMAH[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.4](#).



6.2.24.11 LAN Ethernet MAC Address (LSB) MMAL[n] - 0x0025 + 3*n, n=0...3

This word is loaded by firmware to the 16 LS bits of the MMAL[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.2](#).

6.2.24.12 LAN Ethernet MAC Address (Mid) MMAL[n] - 0x0026 + 3*n, n=0...3

This word is loaded by firmware to the 16 MS bits of the MMAL[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.3](#).

6.2.24.13 LAN Ethernet MAC Address (MSB) MMAH[n] - 0x0027 + 3*n, n=0...3

This word is loaded by firmware to the MMAH[0-3] register.

It is used as the MAC address when a dedicated MAC address mode is used in legacy SMBus.

This is a per-device, per-port value allocated by manufacturing.

For inner structure detail, see [Section 6.2.24.4](#).



6.2.25 PF MAC Address Section Summary Table

Word Offset	Description	Reference
0x0000	Section Header	See section 6.2.25.1
0x0001	PFPM_SAL0	See section 6.2.25.2
0x0002	PFPM_SAL1	See section 6.2.25.3
0x0003	PFPM_SAH0	See section 6.2.25.4
0x0004	PFPM_SAH1	See section 6.2.25.5
0x0005	PFPM_SAL0	See section 6.2.25.6
0x0006	PFPM_SAL1	See section 6.2.25.7
0x0007	PFPM_SAH0	See section 6.2.25.8
0x0008	PFPM_SAH1	See section 6.2.25.9
0x0009	PFPM_SAL0	See section 6.2.25.10
0x000A	PFPM_SAL1	See section 6.2.25.11
0x000B	PFPM_SAH0	See section 6.2.25.12
0x000C	PFPM_SAH1	See section 6.2.25.13
0x000D	PFPM_SAL0	See section 6.2.25.14
0x000E	PFPM_SAL1	See section 6.2.25.15
0x000F	PFPM_SAH0	See section 6.2.25.16
0x0010	PFPM_SAH1	See section 6.2.25.17
0x0011	PFPM_SAL0	See section 6.2.25.18
0x0012	PFPM_SAL1	See section 6.2.25.19
0x0013	PFPM_SAH0	See section 6.2.25.20
0x0014	PFPM_SAH1	See section 6.2.25.21
0x0015	PFPM_SAL0	See section 6.2.25.22
0x0016	PFPM_SAL1	See section 6.2.25.23
0x0017	PFPM_SAH0	See section 6.2.25.24
0x0018	PFPM_SAH1	See section 6.2.25.25
0x0019	PFPM_SAL0	See section 6.2.25.26
0x001A	PFPM_SAL1	See section 6.2.25.27
0x001B	PFPM_SAH0	See section 6.2.25.28
0x001C	PFPM_SAH1	See section 6.2.25.29
0x001D	PFPM_SAL0	See section 6.2.25.30
0x001E	PFPM_SAL1	See section 6.2.25.31
0x001F	PFPM_SAH0	See section 6.2.25.32
0x0020	PFPM_SAH1	See section 6.2.25.33
0x0021	PFPM_SAL0	See section 6.2.25.34
0x0022	PFPM_SAL1	See section 6.2.25.35
0x0023	PFPM_SAH0	See section 6.2.25.36
0x0024	PFPM_SAH1	See section 6.2.25.37



Word Offset	Description	Reference
0x0025	PFPM_SAL0	See section 6.2.25.38
0x0026	PFPM_SAL1	See section 6.2.25.39
0x0027	PFPM_SAH0	See section 6.2.25.40
0x0028	PFPM_SAH1	See section 6.2.25.41
0x0029	PFPM_SAL0	See section 6.2.25.42
0x002A	PFPM_SAL1	See section 6.2.25.43
0x002B	PFPM_SAH0	See section 6.2.25.44
0x002C	PFPM_SAH1	See section 6.2.25.45
0x002D	PFPM_SAL0	See section 6.2.25.46
0x002E	PFPM_SAL1	See section 6.2.25.47
0x002F	PFPM_SAH0	See section 6.2.25.48
0x0030	PFPM_SAH1	See section 6.2.25.49
0x0031	PFPM_SAL0	See section 6.2.25.50
0x0032	PFPM_SAL1	See section 6.2.25.51
0x0033	PFPM_SAH0	See section 6.2.25.52
0x0034	PFPM_SAH1	See section 6.2.25.53
0x0035	PFPM_SAL0	See section 6.2.25.54
0x0036	PFPM_SAL1	See section 6.2.25.55
0x0037	PFPM_SAH0	See section 6.2.25.56
0x0038	PFPM_SAH1	See section 6.2.25.57
0x0039	PFPM_SAL0	See section 6.2.25.58
0x003A	PFPM_SAL1	See section 6.2.25.59
0x003B	PFPM_SAH0	See section 6.2.25.60
0x003C	PFPM_SAH1	See section 6.2.25.61
0x003D	PFPM_SAL0	See section 6.2.25.62
0x003E	PFPM_SAL1	See section 6.2.25.63
0x003F	PFPM_SAH0	See section 6.2.25.64
0x0040	PFPM_SAH1	See section 6.2.25.65

6.2.25.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Block Length		Length in: 2 bytes unit - 1 First Section -> Word: PF MAC Address -> Section Header Last Section -> Word: PF MAC Address -> PFPM_SAH1 Section Length in Words



6.2.25.2 PFPM_SAL0 - 0x0001

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

Bits	Field Name	Default NVM Value	Description
15:0	PFPM_SAL0	0x0000	See respective bits in the PRTPM_SAL register. Note: This field is preserved by the Intel NVM update tool.

6.2.25.3 PFPM_SAL1 - 0x0002

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

Bits	Field Name	Default NVM Value	Description
15:0	PFPM_SAL1	0x0000	See respective bits in the PRTPM_SAL register. Note: This field is preserved by the Intel NVM update tool.

6.2.25.4 PFPM_SAH0 - 0x0003

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

Bits	Field Name	Default NVM Value	Description
15:0	PFPM_SAH0	0x0000	See respective bits in the PRTPM_SAH register. Note: This field is preserved by the Intel NVM update tool.

6.2.25.5 PFPM_SAH1 - 0x0004

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

Bits	Field Name	Default NVM Value	Description
15:0	PFPM_SAH1	0x0000	See respective bits in the PRTPM_SAH register. Note: This field is preserved by the Intel NVM update tool.



6.2.25.6 PFPM_SAL0 - 0x0005

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.7 PFPM_SAL1 - 0x0006

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.8 PFPM_SAH0 - 0x0007

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.9 PFPM_SAH1 - 0x0008

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.10 PFPM_SAL0 - 0x0009

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).



6.2.25.11 PFPM_SAL1 - 0x000A

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.12 PFPM_SAH0 - 0x000B

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.13 PFPM_SAH1 - 0x000C

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.14 PFPM_SAL0 - 0x000D

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.15 PFPM_SAL1 - 0x000E

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).



6.2.25.16 PFPM_SAH0 - 0x000F

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.17 PFPM_SAH1 - 0x0010

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.18 PFPM_SAL0 - 0x0011

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.19 PFPM_SAL1 - 0x0012

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.20 PFPM_SAH0 - 0x0013

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).



6.2.25.21 PFPM_SAH1 - 0x0014

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.22 PFPM_SAL0 - 0x0015

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.23 PFPM_SAL1 - 0x0016

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.24 PFPM_SAH0 - 0x0017

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.25 PFPM_SAH1 - 0x0018

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).



6.2.25.26 PFPM_SAL0 - 0x0019

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.27 PFPM_SAL1 - 0x001A

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.28 PFPM_SAH0 - 0x001B

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.29 PFPM_SAH1 - 0x001C

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.30 PFPM_SAL0 - 0x001D

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).



6.2.25.31 PFPM_SAL1 - 0x001E

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.32 PFPM_SAH0 - 0x001F

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.33 PFPM_SAH1 - 0x0020

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.34 PFPM_SAL0 - 0x0021

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.35 PFPM_SAL1 - 0x0022

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).



6.2.25.36 PFPM_SAH0 - 0x0023

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.37 PFPM_SAH1 - 0x0024

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.38 PFPM_SAL0 - 0x0025

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.39 PFPM_SAL1 - 0x0026

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.40 PFPM_SAH0 - 0x0027

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).



6.2.25.41 PFPM_SAH1 - 0x0028

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.42 PFPM_SAL0 - 0x0029

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.43 PFPM_SAL1 - 0x002A

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.44 PFPM_SAH0 - 0x002B

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

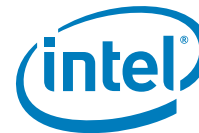
6.2.25.45 PFPM_SAH1 - 0x002C

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).



6.2.25.46 PFPM_SAL0 - 0x002D

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.47 PFPM_SAL1 - 0x002E

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.48 PFPM_SAH0 - 0x002F

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.49 PFPM_SAH1 - 0x0030

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.50 PFPM_SAL0 - 0x0031

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).



6.2.25.51 PFPM_SAL1 - 0x0032

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.52 PFPM_SAH0 - 0x0033

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.53 PFPM_SAH1 - 0x0034

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.54 PFPM_SAL0 - 0x0035

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.55 PFPM_SAL1 - 0x0036

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).



6.2.25.56 PFPM_SAH0 - 0x0037

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.57 PFPM_SAH1 - 0x0038

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.58 PFPM_SAL0 - 0x0039

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.59 PFPM_SAL1 - 0x003A

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.60 PFPM_SAH0 - 0x003B

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).



6.2.25.61 PFPM_SAH1 - 0x003C

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).

6.2.25.62 PFPM_SAL0 - 0x003D

This word is loaded by firmware to the appropriate PRTPM_SAL register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.2](#).

6.2.25.63 PFPM_SAL1 - 0x003E

This register is loaded by firmware to the appropriate PRTPM_SAL register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.3](#).

6.2.25.64 PFPM_SAH0 - 0x003F

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [1,0].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.4](#).

6.2.25.65 PFPM_SAH1 - 0x0040

This register is loaded by firmware to the appropriate PRTPM_SAH register, bytes [3:2].

One MAC address per enabled PF, and up to 4 PFs per ports.

This is a per-device, per-enabled PF value allocated by manufacturing.

For inner structure detail, see [Section 6.2.25.5](#).



6.2.26 PHY Capability Data Structure 0 Section Summary Table

The PHY Capabilities Data structure contains all the parameters to control the internal and external PHY operation. For example, interface type and mode, EEE parameters, etc.

Data structure is repeated per port.

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.26.1
0x0001	INT-EXT PHY Select	See section 6.2.26.2
0x0002	Internal PHY Type	See section 6.2.26.3
0x0003	External PHY Type	See section 6.2.26.4
0x0004	PHY ID 0	See section 6.2.26.5
0x0005	PHY ID 1	See section 6.2.26.6
0x0006	Module Type 0	See section 6.2.26.7
0x0007	Module Type 1	See section 6.2.26.8
0x0008	PHY Capabilities Misc0	See section 6.2.26.9
0x0009	PHY Capabilities Misc1	See section 6.2.26.10
0x000A	40 LESM Timer Values	See section 6.2.26.11
0x000B	PHY Capabilities Misc2	See section 6.2.26.12
0x000C	PHY Capabilities Misc3	See section 6.2.26.13
0x000D	General Timer Values	

6.2.26.1 Section Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in: 2 bytes unit - 1 First Section -> Word: PHY Capability Data Structure 0 -> Section Length Last Section -> Word: PHY Capability Data Structure 0 -> General Timer Values

6.2.26.2 INT-EXT PHY Select - 0x0001

Bits	Field Name	Default NVM Value	Description
15:9	Reserved	0x0	Reserved.
8	X710-TM4 Mode	0x0	0b = The X710-TM4 is not used. 1b = The X710-TM4 is used. For internal PHY firmware download indication and to distinguish between 10GBASE-T and the X710-TM4.
7	External PHY Mode	0x0	0b = External PHY is not used. 1b = External PHY is used for external 25 GbE interface. Note: This field is preserved by the Intel NVM update tool.
6	Reserved	0x0	Reserved.
5	EN_1G_LPLU	0x1	Note: This field is preserved by the Intel NVM update tool.



Bits	Field Name	Default NVM Value	Description
4	Reserved	0x0	Reserved.
3:0	Int_Ext PHY	0x0	Internal/External PHY Select. 0x0 = Internal PHY only. 0x1 = External 10GBASE-T PHY. 0x2 = External SFP+ module. 0x3 = Reserved. 0x4 = Reserved. Note: This field is preserved by the Intel NVM update tool.

INT-EXT PHY configuration should be as follows:

Setup	X710-TM4 Mode	Int_Ext PHY
BASE-T ports internal PHY (port 0/2)	0x1	0x1
Non-BASE-T ports in BP mode (port 1/3)	0x0	0x0
Non-BASE-T ports connected to SFP cages (port 1/3)	0x0	0x2
BASE-T ports external PHY (port 1/3)	0x1	0x1

6.2.26.3 Internal PHY Type - 0x0002

This parameter is used by firmware to find out the various internal PHY types to be supported on a port. The device might support multiple PHY types on the same port.

One of the PHY types might be enabled due to the result of auto-negotiation or manually set by firmware when auto-negotiation is disabled or not used.

One bit per PHY type.

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	Note: This field is preserved by the Intel NVM update tool.
7	SFI	0x0	Note: This field is preserved by the Intel NVM update tool.
5:4	Reserved	0x0	Note: This field is preserved by the Intel NVM update tool.
3	1000BASE-KR	0x0	Note: This field is preserved by the Intel NVM update tool.
2:1	Reserved	0x0	Note: This field is preserved by the Intel NVM update tool.
0	SGMII	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.26.4 External PHY Type - 0x0003

This parameter is used by firmware to find out the various external PHY types to be supported on a port. The device might support multiple PHY types on the same port.

One of the PHY types might be enabled due to the result of auto-negotiation or manually set by firmware when auto-negotiation is disabled or not used.

One bit per PHY type.

Bits	Field Name	Default NVM Value	Description
15:7	Reserved	0x0	Note: This field is preserved by the Intel NVM update tool.



Bits	Field Name	Default NVM Value	Description
6	10GBASE-SFP+	0x0	Note: This field is preserved by the Intel NVM update tool.
5	10GBASE-LR	0x0	Note: This field is preserved by the Intel NVM update tool.
4	10GBASE-SR	0x0	Note: This field is preserved by the Intel NVM update tool.
3	10GBASE-T	0x0	Note: This field is preserved by the Intel NVM update tool.
2	1000BASE-T	0x0	Note: This field is preserved by the Intel NVM update tool.
1	100BASE-TX	0x0	Note: This field is preserved by the Intel NVM update tool.
0	1000BASE-T SFP	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.26.5 PHY ID 0 - 0x0004

This parameter is used by firmware to verify the 10GBASE-T PHY-ID connected on a port.

The format of the PHY ID is defined in IEEE Std 802.3 and contains OUI, manufacturers model number and revision number of the PHY.

Bits	Field Name	Default NVM Value	Description
15:8	PHY ID Byte1	0x0	Note: This field is preserved by the Intel NVM update tool.
7:0	PHY ID Byte0	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.26.6 PHY ID 1 - 0x0005

This parameter is used by firmware to verify the 10GBASE-T PHY-ID connected on a port.

The format of the PHY ID is defined in IEEE Std 802.3 and contains OUI, manufacturers model number and revision number of the PHY.

Bits	Field Name	Default NVM Value	Description
15:8	PHY ID Byte3	0x0	Note: This field is preserved by the Intel NVM update tool.
7:0	PHY ID Byte2	0x0	Note: This field is preserved by the Intel NVM update tool.



6.2.26.7 Module Type 0 - 0x0006

This parameter is used by firmware to find out the module type on a port when connected to external modules. For example, the X710-TM4 might be connected to an SFP+ optical or direct attached copper modules. The format of the module type returns the ID and extended ID fields as defined in the SFP+ specifications.

Bits	Field Name	Default NVM Value	Description
15:8	10G Compliance Code	0x0	<p>The SFF standard defines an 8-byte field used to indicate the electrical or optical support modes. These are bit-significant indicators that define the electronic or optical interfaces that are supported by the transceiver. At least one bit must be set in this field.</p> <p>The SFP standard defines this field in bytes 3-10 of the standard address space.</p> <p>For 10 GbE SFP+ modules this byte is decoded as follows: Bit 0 = SFP+ Cu passive. Bit 1 = SFP+ Cu active. Bit 2 = Reserved. Bit 3 = Reserved. Bits 7:4 = Decoded according to the SFP+ 10G compliance codes (byte 3). Bit 4 = 10GBASE-SR. Bit 5 = 10GBASE-LR. Bit 6 = Reserved. Bit 7 = Reserved.</p> <p>Note: This field is preserved by the Intel NVM update tool.</p>
7:0	Module Identifier	0x3	<p>Defined by SFP+ (address 0xA0, byte 0) specifications. 0x3 = SFP/SFP28. 0xD = Reserved. Valid values are: 0x3 = SFP+/SFP28. 0xD = Reserved.</p> <p>Note: This field is preserved by the Intel NVM update tool.</p> <p>Valid values are: 0x3 = SFP+. 0xD = Reserved.</p> <p>Note: This field is preserved by the Intel NVM update tool.</p>



6.2.26.8 Module Type 1 - 0x0007

This parameter is used by firmware to find out the module type on a port when connected to external modules. For example, the X710-TM4 might be connected to an SFP+ optical or direct attached copper modules. The format of the module type returns the ID and extended ID fields as defined in the SFP+ specifications.

Bits	Field Name	Default NVM Value	Description
15:10	Reserved	0x0	Reserved.
9:8	Pause Ability Default	0x0	<p>This parameter is used by firmware to control the default setting for IEEE Std. 802.3x pause ability of a port.</p> <p>This default is also advertised if auto-negotiation is enabled.</p> <p>Set bit 8 to 1b to enable IEEE Std. 802.3x Tx link pause ability or set to 0b to disable link pause ability.</p> <p>Set bit 9 to 1b to enable IEEE Std. 802.3x Rx link pause ability or set to 0b to disable link pause ability.</p> <p>Note: This field is preserved by the Intel NVM update tool.</p>
7:0	1G Compliance Code	0x0	<p>The SFF standard defines an 8-byte field used to indicate the electrical or optical support modes.</p> <p>These are bit-significant indicators that define the electronic or optical interfaces that are supported by the transceiver. At least one bit must be set in this field.</p> <p>The SFP standard defines this field in byte 3-10 of the standard address space.</p> <p>1G compliance codes are in SFP byte 6.</p> <p>Note: This field is preserved by the Intel NVM update tool.</p>



6.2.26.9 PHY Capabilities Misc0 - 0x0008

Bits	Field Name	Default NVM Value	Description
15:13	Reserved	0x0	Reserved
12	Disable Firmware Link Management	0x0	This bit is used to disable the firmware's link management and enable direct software control of the link parameters for debug. Note: This field is preserved by the Intel NVM update tool.
11	Enable Module Qualification	0x0	When, set to 1b, the module qualification process is enabled. Note: This field is preserved by the Intel NVM update tool.
10	Low Power Ability	0x1	Low Power Ability. 1b = Low power. 0b = High power. This field controls the ability and has a different meaning for BASE-T. BASE-T: This field indicates if the PHY can be placed into a low power state. Note: This disables the link. When power ability equals low power, it means that the software device driver might set the BASE-T PHY to low power mode and disable the link. Should be set to low power by default.
9:8	Pause Ability	0x3	This parameter is used by firmware to configure or advertise the IEEE Std. 802.3x pause ability of a port Note: The Link Flow Control (LFC or Link Pause) is enabled through pause ability bits during PHY auto-negotiation and the PFC is enabled through DCBX protocol. The LFC and PFC features are mutually exclusive. So management should disable link pause on links where PFC should be enabled. Bit 0.0 = Set to 1b to enable IEEE Std. 802.3x Tx link pause ability, or set to 0b to disable link pause ability. Bit 0.1 = Set to 1b to enable IEEE Std. 802.3x Rx link pause ability, or set to 0b to disable link pause ability. Note: This field is preserved by the Intel NVM update tool.
7:0	Link Speed	0x8	This parameter is used by firmware to configure or advertise various link speeds supported on a port. The device might have the ability to support multiple link speeds on the same port. One of the link speeds might be enabled due to the result of auto-negotiation or manually set by firmware when auto-negotiation is disabled or not used. Link rate supported on a port: Bit 0.0 - 2.5 Gb/s. Bit 0.1 - 100 Mb/s. Bit 0.2 - 1 Gb/s. Bit 0.3 - 10 Gb/s. Bit 0.4 - Reserved. Bit 0.5 - Reserved. Bit 0.6 - Reserved. Bit 0.7 = 5 Gb/s. Other bits = Reserved. Must be zero. Note: This field is preserved by the Intel NVM update tool.

Note: When X710-TM4 mode is enabled with 2.5 GbE/5 GbE speeds, bit 0.3 (10 GbE) must also be set in the NVM. Firmware should report the link speed to software based only on the external (line-side) link speed.



6.2.26.10 PHY Capabilities Misc1 - 0x0009

Bits	Field Name	Default NVM Value	Description
15:8	Multi-Speed Module Timeout	0x64	Configurable number of ms to wait for link establishment when running the multi-speed module flow. Note: This field is preserved by the Intel NVM update tool.
7:0	EEE Capability	0x40	This parameter is used by firmware to configure the EEE capability of various PHY types supported on a port. The device might support multiple PHY types on the same port and EEE capability is indicated for each PHY type. One bit per PHY type. The device might be capable of supporting multiple PHY types. The following parameter indicates the bit number. Bit 0.0 = EEE native/autonomous mode. This bit is only relevant for BASE-T PHYs: 0b = Native mode. 1b = Autonomous mode. Bit 0.1 = EEE is supported for 100BASE-TX. Bit 0.2 = EEE is supported for 1000BASE-T. Bit 0.3 = EEE is supported for 10/5/2.5GBASE-T. Bit 0.4 = Reserved. Bit 0.5 = Reserved. Bit 0.6 = EEE is supported for 10GBASE-KR. Note: This field is preserved by the Intel NVM update tool.

Note: When in X710-TM4 mode is enabled, EEE for 1 GbE and 100 Mb/s must be the same.

6.2.26.11 40 LESM Timer Values - 0x000A

40 LESM timer values.

Bits	Field Name	Default NVM Value	Description
15:8	40_LESM_TIMER1	0x0A	Time to wait for link to be established. Configured in 0.1 seconds. Note: This field is preserved by the Intel NVM update tool.
7:0	40_LESM_TIMER0	0x1E	Time to wait for link to be established. Configured in 0.1 seconds. Note: This field is preserved by the Intel NVM update tool.



6.2.26.12 PHY Capabilities Misc2 - 0x000B

Bits	Field Name	Default NVM Value	Description
15:10	Module Temp Support	0x0	
9:8	Reserved	0x0	Reserved.
7:2	PE_RST_N Link Delay Timer	0x0	Delay to add after PE_RST_N assertion to start the link-monitor flow. Firmware adds this delay value (each unit is 100 ms) to the basic 500 ms delay from PE_RST_N, and only when the total delay expired, firmware starts the link-monitor flow. This is done to prevent corruption of the SFP modules due to a truncated transaction before power down. Note: This field is preserved by the Intel NVM update tool.
1	NP ANEG Disable	0x1	NP ANEG Disable. 0b = Next page auto-negotiation is enabled based on capabilities. 1b = Perform ANEG without next page capabilities. For example, EEE to enable next page capabilities only if LP supports next page. 0x0 = Enabled. Next page auto-negotiation is enabled based on capabilities. 0x1 = Disabled. Perform ANEG without next page capabilities. Note: This field is preserved by the Intel NVM update tool.
0	Default Low Power Ability	0x0	Default Low Power Ability. 1b = Low power mode. 0b = High power mode. This is the default value that the system operates in. It has a different meaning for BASE-T. For BASE-T, this is ignored if the NVM loaded low power ability equals high power. Also, this field should be set to high power mode for the link to come up at power up. Note: This field is preserved by the Intel NVM update tool.

6.2.26.13 PHY Capabilities Misc3 - 0x000C

Bits	Field Name	Default NVM Value	Description
15:11	Reserved	0x0	Reserved. Note: This field is preserved by the Intel NVM update tool.
10	Extended External PHY Type 2.5GBASE-T	0x0	Note: This field is preserved by the Intel NVM update tool.
9	Extended External PHY Type 5GBASE-T	0x0	Note: This field is preserved by the Intel NVM update tool.
8:0	Reserved	0x0	Reserved. Note: This field is preserved by the Intel NVM update tool.

6.2.26.14 General Timer Values - 0x000D

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	Reserved.
7:0	Link Remote Fault Timeout	0x32	This is the timeout value for resetting the link state machine if remote faults are received for this amount of time. Increments are in steps of 100 ms. Note: This field is preserved by the Intel NVM update tool.



6.2.27 Feature Configuration Section Summary Table

NVM adaptive feature configuration.

Word Offset	Description	Reference

6.2.28 Immediate Values Section Summary Table

Required for NVM update with existing NUP.

Word Offset	Description	Reference

6.2.29 External to Internal PHY Mapping 0 Section Summary Table

This section provides the internal PHY mode to be selected for each external PHY/Module counter part.

The X710-TM4 provides multiple PHY interfaces for connecting different types of external PHYs and optical or copper modules. Users can plug-in different types of modules into the SFP+ connectors and choose from several connectivity options when using external PHYs.

Firmware has to read the PHY or module ID and choose the appropriate interface to operate with that external module. Further, when operating with an external 10GBASE-T PHY auto-negotiation might result in several speed modes. For example, when a 10GBASE-T PHY can auto-negotiate to 1 GbE. In this case firmware needs to reduce the internal link to SGMII.

Note: The mapping words define the interface between the X710-TM4 and the internal PHY host side for the relevant 2 ports, and the MCP interface of the other 2 ports when enabled.

Supported configurations are KR/XFI for 10 GbE/5 GbE/2.5 GbE; SGMII for 1 GbE and SGMII for 100 Mb/s.

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.29.1
0x0001	Mapping Word0	See section 6.2.29.2
0x0002	Mapping Word1	See section 6.2.29.3
0x0003	Mapping Word2	See section 6.2.29.4



6.2.29.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in: 2 bytes unit - 1 First Section -> Word: External to Internal PHY Mapping 0 -> Section Length Last Section -> Word: External to Internal PHY Mapping 0 -> Mapping Word2



6.2.29.2 Mapping Word0 - 0x0001

Bits	Field Name	Default NVM Value	Description
15:14	Reserved	0x0	Reserved. Note: This field is preserved by the Intel NVM update tool.
13:12	10GBASE-SFP+Cu	0x0	Internal interface for a 10GBASE-SFP+Cu external module. 0x0 = Reserved. 0x1 = SFI. 0x2 = Reserved. Other values = Reserved. Note: This field is preserved by the Intel NVM update tool.
11:10	10GBASE-LR	0x0	Internal interface for a 10GBASE-LR external module. 2 bit field [0..1] 0x0 = Reserved. 0x1 = SFI. 0x2 = KR. Other values = Reserved. Note: This field is preserved by the Intel NVM update tool.
9:8	10GBASE-SR	0x0	Internal interface for a 10GBASE-SR external module. 2 bit field [0..1] 0x0 = Reserved. 0x1 = SFI. 0x2 = KR. Other values = Reserved. Note: This field is preserved by the Intel NVM update tool.
7:6	10GBASE-T	0x0	Internal interface for a 10GBASE-TX external PHY. 0x0 = Reserved. 0x1 = 10GBASE-KR. 0x2 = SFI. 0x3 = Reserved. Note: This field is preserved by the Intel NVM update tool.
5:4	1000BASE-T	0x0	Internal interface for a 1000BASE-T external PHY. 0x0 = SGMII. 0x2 = KX. Other values = Reserved. Note: Note: This field is preserved by the Intel NVM update tool.
3:2	100BASE-TX	0x0	Internal interface for a 100BASE-TX external PHY. 0x0 = SGMII. Other values = Reserved. Note: This field is preserved by the Intel NVM update tool.
1:0	Reserved	0x0	Reserved. Note: This field is preserved by the Intel NVM update tool.

6.2.29.3 Mapping Word1 - 0x0002

Bits	Field Name	Default NVM Value	Description
15:10	Reserved	0x1	Reserved. Note: This field is preserved by the Intel NVM update tool.
7:6	1000BASE-T SFP	0x0	Internal interface for a 1GBASE-T SFP external module. 0x0 = Reserved. 0x1 = Reserved. 0x2 = SGMII with Clause 37 auto-negotiation enabled. 0x3 = Reserved. Note: This field is preserved by the Intel NVM update tool.



Bits	Field Name	Default NVM Value	Description
5:0	Reserved	0x0	Reserved. Note: This field is preserved by the Intel NVM update tool.

6.2.29.4 Mapping Word2 - 0x0003

Bits	Field Name	Default NVM Value	Description
15:2	Reserved	0x0	Reserved.
1:0	Reserved	0x1	Reserved. Note: This field is preserved by the Intel NVM update tool.

6.2.30 Current Settings Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.30.1
0x0001	Current LLDP Admin Status	See section 6.2.30.2

6.2.30.1 Section Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in: 2 bytes unit - 1 First Section -> Word: Current Settings -> Section Length Last Section -> Word: Current Settings -> Current LLDP Admin Status

6.2.30.2 Current LLDP Admin Status - 0x0001

Bits	Field Name	Default NVM Value	Description
15:12	Port 3	0xF	Defines status of LLDP agent. Applies to LAN Port 3.
11:8	Port 2	0xF	Defines status of LLDP agent. Applies to LAN Port 2.
7:4	Port 1	0xF	Defines status of LLDP agent. Applies to LAN Port 1.
3:0	Port 0	0xF	Defines status of LLDP agent. Applies to LAN Port 0.

6.2.31 Auto Generated Pointers Module Section Summary Table

Pointers to Type 1/2 words used by the EMP and software.

Word Offset	Description	Reference
0x7D80	Module Length	See section 6.2.31.1
0x7D81	Pointer to PFPM_APM Section	See section 6.2.31.2
0x7D82	Pointer to PFPM_APM Offset	See section 6.2.31.3
0x7D83	Pointer to PRTPM_GC Section	See section 6.2.31.4
0x7D84	Pointer to PRTPM_GC Offset	See section 6.2.31.5
0x7D85	Pointer to GLGEN_STAT Section	See section 6.2.31.6



Word Offset	Description	Reference
0x7D86	Pointer to GLGEN_STAT Offset	See section 6.2.31.7
0x7D87	Pointer to GLPCI_SERL Section	See section 6.2.31.8
0x7D88	Pointer to GLPCI_SERL Offset	See section 6.2.31.9
0x7D89	Pointer to GLPCI_SERH Section	See section 6.2.31.10
0x7D8A	Pointer to GLPCI_SERH Offset	See section 6.2.31.11
0x7D8B	Pointer to PRTGL_SAL Section	See section 6.2.31.12
0x7D8C	Pointer to PRTGL_SAL Offset	See section 6.2.31.13
0x7D8D	Pointer to PRTGL_SAH Section	See section 6.2.31.14
0x7D8E	Pointer to PRTGL_SAH Offset	See section 6.2.31.15
0x7D8F	Pointer to GLPCI_CAPSUP Section	See section 6.2.31.16
0x7D90	Pointer to GLPCI_CAPSUP Offset	See section 6.2.31.17
0x7D91	Pointer to PRTDCB_MFLCN Section	See section 6.2.31.18
0x7D92	Pointer to PRTDCB_MFLCN Offset	See section 6.2.31.19
0x7D93	Pointer to PRTDCB_FCCFG Section	See section 6.2.31.20
0x7D94	Pointer to PRTDCB_FCCFG Offset	See section 6.2.31.21
0x7D95	Pointer to PFGEN_PORTNUM Section	See section 6.2.31.22
0x7D96	Pointer to PFGEN_PORTNUM Offset	See section 6.2.31.23
0x7D97	Pointer to PFPCI_FUNC2 Section	See section 6.2.31.24
0x7D98	Pointer to PFPCI_FUNC2 Offset	See section 6.2.31.25
0x7D99	Pointer to PFPCI_CLASS Section	See section 6.2.31.26
0x7D9A	Pointer to PFPCI_CLASS Offset	See section 6.2.31.27
0x7D9B	Pointer to PF_VT_PFALLOC_PCIE Section	See section 6.2.31.28
0x7D9C	Pointer to PF_VT_PFALLOC_PCIE Offset	See section 6.2.31.29
0x7D9D	Pointer to PF_VT_PFALLOC Section	See section 6.2.31.30
0x7D9E	Pointer to PF_VT_PFALLOC Offset	See section 6.2.31.31
0x7D9F	Pointer to GLGEN_PCIFCNCNT Section	See section 6.2.31.32
0x7DA0	Pointer to GLGEN_PCIFCNCNT Offset	See section 6.2.31.33
0x7DA1	Pointer to GLPCI_REVID Section	See section 6.2.31.34
0x7DA2	Pointer to GLPCI_REVID Offset	See section 6.2.31.35
0x7DA3	Pointer to PFPCI_DEVID Section	See section 6.2.31.36
0x7DA4	Pointer to PFPCI_DEVID Offset	See section 6.2.31.37
0x7DA5	Pointer to GLPCI_SUBVENID Section	See section 6.2.31.38
0x7DA6	Pointer to GLPCI_SUBVENID Offset	See section 6.2.31.39
0x7DA7	Pointer to PFPCI_SUBSYSID Section	See section 6.2.31.40
0x7DA8	Pointer to PFPCI_SUBSYSID Offset	See section 6.2.31.41
0x7DA9	Pointer to GLPCI_VENDORID Section	See section 6.2.31.42
0x7DAA	Pointer to GLPCI_VENDORID Offset	See section 6.2.31.43
0x7DAB	Pointer to GLPCI_CNF2 Section	See section 6.2.31.44
0x7DAC	Pointer to GLPCI_CNF2 Offset	See section 6.2.31.45



Word Offset	Description	Reference
0x7DAD	Pointer to PFPCI_FUNC Section	See section 6.2.31.46
0x7DAE	Pointer to PFPCI_FUNC Offset	See section 6.2.31.47
0x7DAF	Pointer to PRTMAC_HSEC_CTL_TX_SA_PART1 Section	See section 6.2.31.48
0x7DB0	Pointer to PRTMAC_HSEC_CTL_TX_SA_PART1 Offset	See section 6.2.31.49
0x7DB1	Pointer to PRTMAC_HSEC_CTL_TX_SA_PART2 Section	See section 6.2.31.50
0x7DB2	Pointer to PRTMAC_HSEC_CTL_TX_SA_PART2 Offset	See section 6.2.31.51
0x7DB3	Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 Section	See section 6.2.31.52
0x7DB4	Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 Offset	See section 6.2.31.53
0x7DB5	Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 Section	See section 6.2.31.54
0x7DB6	Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 Offset	See section 6.2.31.55
0x7DB7	Pointer to GLPCI_CAPCTRL Section	See section 6.2.31.56
0x7DB8	Pointer to GLPCI_CAPCTRL Offset	See section 6.2.31.57
0x7DB9	Pointer to GLGEN_GPIO_CTL Section	See section 6.2.31.58
0x7DBA	Pointer to GLGEN_GPIO_CTL Offset	See section 6.2.31.59
0x7DBB	Pointer to GLGEN_LED_CTL Section	See section 6.2.31.60
0x7DBC	Pointer to GLGEN_LED_CTL Offset	See section 6.2.31.61
0x7DBD	Pointer to PFGEN_PORTNUM_CAR Section	See section 6.2.31.62
0x7DBE	Pointer to PFGEN_PORTNUM_CAR Offset	See section 6.2.31.63
0x7DBF	Pointer to PFGEN_PORTMDIO_NUM Section	See section 6.2.31.64
0x7DC0	Pointer to PFGEN_PORTMDIO_NUM Offset	See section 6.2.31.65
0x7DC1	Pointer to PRTTSYN_CTL0 Section	See section 6.2.31.66
0x7DC2	Pointer to PRTTSYN_CTL0 Offset	See section 6.2.31.67
0x7DC3	Pointer to GLFOC_CACHE_CTL Section	See section 6.2.31.68
0x7DC4	Pointer to GLFOC_CACHE_CTL Offset	See section 6.2.31.69
0x7DC5	Pointer to PFQF_CTL_0 Section	See section 6.2.31.70
0x7DC6	Pointer to PFQF_CTL_0 Offset	See section 6.2.31.71



6.2.31.1 Module Length - 0x7D80

Bits	Field Name	Default NVM Value	Description
15:0	Module Length		Length in: 2 bytes unit - 1 First Section -> Word: Auto Generated Pointers Module -> Module Length Last Section -> Word: Auto Generated Pointers Module -> Pointer to PFQF_CTL_0 Offset

6.2.31.2 Pointer to PFPM_APM Section - 0x7D81

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFPM_APM Section	0x0	

6.2.31.3 Pointer to PFPM_APM Offset - 0x7D82

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFPM_APM Offset	0xB8	

6.2.31.4 Pointer to PRTPM_GC Section - 0x7D83

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRTPM_GC Section		

6.2.31.5 Pointer to PRTPM_GC Offset - 0x7D84

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTPM_GC Offset	0xE6	

6.2.31.6 Pointer to GLGEN_STAT Section - 0x7D85

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLGEN_STAT Section	0x0	

6.2.31.7 Pointer to GLGEN_STAT Offset - 0x7D86

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLGEN_STAT Offset	0x17	



6.2.31.8 Pointer to GLPCI_SERL Section - 0x7D87

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_SERL Section	0x0	

6.2.31.9 Pointer to GLPCI_SERL Offset - 0x7D88

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_SERL Offset	0xDF	

6.2.31.10 Pointer to GLPCI_SERH Section - 0x7D89

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLPCI_SERH Section		

6.2.31.11 Pointer to GLPCI_SERH Offset - 0x7D8A

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_SERH Offset	0xE1	

6.2.31.12 Pointer to PRTGL_SAL Section - 0x7D8B

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTGL_SAL Section	0x0	

6.2.31.13 Pointer to PRTGL_SAL Offset - 0x7D8C

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTGL_SAL Offset	0x36	

6.2.31.14 Pointer to PRTGL_SAH Section - 0x7D8D

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRTGL_SAH Section		



6.2.31.15 Pointer to PRTGL_SAH Offset - 0x7D8E

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTGL_SAH Offset	0x41	

6.2.31.16 Pointer to GLPCI_CAPSUP Section - 0x7D8F

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLPCI_CAPSUP Section		

6.2.31.17 Pointer to GLPCI_CAPSUP Offset - 0x7D90

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_CAPSUP Offset	0xE8	

6.2.31.18 Pointer to PRTDCB_MFLCN Section - 0x7D91

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRTDCB_MFLCN Section		

6.2.31.19 Pointer to PRTDCB_MFLCN Offset - 0x7D92

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTDCB_MFLCN Offset	0x4C	

6.2.31.20 Pointer to PRTDCB_FCCFG Section - 0x7D93

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRTDCB_FCCFG Section		

6.2.31.21 Pointer to PRTDCB_FCCFG Offset - 0x7D94

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTDCB_FCCFG Offset	0x1C1	



6.2.31.22 Pointer to PFGEN_PORTNUM Section - 0x7D95

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFGEN_PORTNUM Section	0x0	

6.2.31.23 Pointer to PFGEN_PORTNUM Offset - 0x7D96

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFGEN_PORTNUM Offset	0x221	

6.2.31.24 Pointer to PFPCI_FUNC2 Section - 0x7D97

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PFPCI_FUNC2 Section		

6.2.31.25 Pointer to PFPCI_FUNC2 Offset - 0x7D98

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFPCI_FUNC2 Offset	0x6C	

6.2.31.26 Pointer to PFPCI_CLASS Section - 0x7D99

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PFPCI_CLASS Section		

6.2.31.27 Pointer to PFPCI_CLASS Offset - 0x7D9A

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFPCI_CLASS Offset	0xB2	

6.2.31.28 Pointer to PF_VT_PFALLOC_PCIE Section - 0x7D9B

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PF_VT_PFALLOC_PCIE Section		



6.2.31.29 Pointer to PF_VT_PFALLOC_PCIE Offset - 0x7D9C

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PF_VT_PFALLOC_PCIE Offset	0x8F	

6.2.31.30 Pointer to PF_VT_PFALLOC Section - 0x7D9D

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PF_VT_PFALLOC Section		

6.2.31.31 Pointer to PF_VT_PFALLOC Offset - 0x7D9E

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PF_VT_PFALLOC Offset	0x244	

6.2.31.32 Pointer to GLGEN_PCIFCNCNT Section - 0x7D9F

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLGEN_PCIFCNCNT Section		

6.2.31.33 Pointer to GLGEN_PCIFCNCNT Offset - 0x7DA0

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLGEN_PCIFCNCNT Offset	0x100	

6.2.31.34 Pointer to GLPCI_REVID Section - 0x7DA1

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLPCI_REVID Section		

6.2.31.35 Pointer to GLPCI_REVID Offset - 0x7DA2

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_REVID Offset	0xEE	



6.2.31.36 Pointer to PFPCI_DEVID Section - 0x7DA3

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PFPCI_DEVID Section		

6.2.31.37 Pointer to PFPCI_DEVID Offset - 0x7DA4

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFPCI_DEVID Offset	0x26	

6.2.31.38 Pointer to GLPCI_SUBVENID Section - 0x7DA5

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLPCI_SUBVENID Section		

6.2.31.39 Pointer to GLPCI_SUBVENID Offset - 0x7DA6

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_SUBVENID Offset	0xD9	

6.2.31.40 Pointer to PFPCI_SUBSYSID Section - 0x7DA7

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PFPCI_SUBSYSID Section		

6.2.31.41 Pointer to PFPCI_SUBSYSID Offset - 0x7DA8

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFPCI_SUBSYSID Offset	0x49	

6.2.31.42 Pointer to GLPCI_VENDORID Section - 0x7DA9

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLPCI_VENDORID Section		



6.2.31.43 Pointer to GLPCI_VENDORID Offset - 0x7DAA

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_VENDORID Offset	0xFC	

6.2.31.44 Pointer to GLPCI_CNF2 Section - 0x7DAB

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLPCI_CNF2 Section		

6.2.31.45 Pointer to GLPCI_CNF2 Offset - 0x7DAC

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_CNF2 Offset	0xDD	

6.2.31.46 Pointer to PFPCI_FUNC Section - 0x7DAD

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PFPCI_FUNC Section		

6.2.31.47 Pointer to PFPCI_FUNC Offset - 0x7DAE

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFPCI_FUNC Offset	0x10E	

6.2.31.48 Pointer to PRTMAC_HSEC_CTL_TX_SA_PART1 Section - 0x7DAF

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRTMAC_HSEC_CTL_TX_SA_PART1 Section		

6.2.31.49 Pointer to PRTMAC_HSEC_CTL_TX_SA_PART1 Offset - 0x7DB0

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTMAC_HSEC_CTL_TX_SA_PART1 Offset	0x13B	



6.2.31.50 Pointer to PRTMAC_HSEC_CTL_TX_SA_PART2 Section - 0x7DB1

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRTMAC_HSEC_CTL_TX_SA_PART2 Section		

6.2.31.51 Pointer to PRTMAC_HSEC_CTL_TX_SA_PART2 Offset - 0x7DB2

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTMAC_HSEC_CTL_TX_SA_PART2 Offset	0x142	

6.2.31.52 Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 Section - 0x7DB3

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 Section		

6.2.31.53 Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 Offset - 0x7DB4

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1 Offset	0x85	



6.2.31.54 Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 Section - 0x7DB5

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 Section		

6.2.31.55 Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 Offset - 0x7DB6

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2 Offset	0x8C	

6.2.31.56 Pointer to GLPCI_CAPCTRL Section - 0x7DB7

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLPCI_CAPCTRL Section		

6.2.31.57 Pointer to GLPCI_CAPCTRL Offset - 0x7DB8

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLPCI_CAPCTRL Offset	0xE6	

6.2.31.58 Pointer to GLGEN_GPIO_CTL Section - 0x7DB9

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLGEN_GPIO_CTL Section		

6.2.31.59 Pointer to GLGEN_GPIO_CTL Offset - 0x7DBA

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLGEN_GPIO_CTL Offset	0x2A	



6.2.31.60 Pointer to GLGEN_LED_CTL Section - 0x7DBB

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLGEN_LED_CTL Section		

6.2.31.61 Pointer to GLGEN_LED_CTL Offset - 0x7DBC

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLGEN_LED_CTL Offset	0x66	

6.2.31.62 Pointer to PFGEN_PORTNUM_CAR Section - 0x7DBD

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PFGEN_PORTNUM_CAR Section		

6.2.31.63 Pointer to PFGEN_PORTNUM_CAR Offset - 0x7DBE

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFGEN_PORTNUM_CAR Offset	0x95	

6.2.31.64 Pointer to PFGEN_PORTMDIO_NUM Section - 0x7DBF

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PFGEN_PORTMDIO_NUM Section		

6.2.31.65 Pointer to PFGEN_PORTMDIO_NUM Offset - 0x7DC0

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFGEN_PORTMDIO_NUM Offset	0x3	

6.2.31.66 Pointer to PRRTSYN_CTL0 Section - 0x7DC1

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PRRTSYN_CTL0 Section		



6.2.31.67 Pointer to PRTTSYN_CTL0 Offset - 0x7DC2

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PRTTSYN_CTL0 Offset	0x162	

6.2.31.68 Pointer to GLFOC_CACHE_CTL Section - 0x7DC3

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to GLFOC_CACHE_CTL Section		

6.2.31.69 Pointer to GLFOC_CACHE_CTL Offset - 0x7DC4

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to GLFOC_CACHE_CTL Offset	0xD6	

6.2.31.70 Pointer to PFQF_CTL_0 Section - 0x7DC5

Bits	Field Name	Default NVM Value	Description
15:0	Cloned Pointer to PFQF_CTL_0 Section		

6.2.31.71 Pointer to PFQF_CTL_0 Offset - 0x7DC6

Bits	Field Name	Default NVM Value	Description
15:0	Pointer to PFQF_CTL_0 Offset	0x2A1	

6.2.32 PCIe Analog Section Summary Table

Configures the analog section of the PCIe PHY.

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.32.1
0x0001	PCIe Analog Data	See section 6.2.32.2
0x0002	moduleTypeL	See section 6.2.32.3
0x0003	moduleTypeH	See section 6.2.32.4
0x0004	headerLenL	See section 6.2.32.5
0x0005	headerLenH	See section 6.2.32.6
0x0006	headerVersionL	See section 6.2.32.7
0x0007	headerVersionH	See section 6.2.32.8



Word Offset	Description	Reference
0x0008	moduleIDL	See section 6.2.32.9
0x0009	moduleIDH	See section 6.2.32.10
0x000A	moduleVendorL	See section 6.2.32.11
0x000B	moduleVendorH	See section 6.2.32.12
0x000C	dateL	See section 6.2.32.13
0x000D	dateH	See section 6.2.32.14
0x000E	sizeL	See section 6.2.32.15
0x000F	sizeH	See section 6.2.32.16
0x0010	keySizeL	See section 6.2.32.17
0x0011	keySizeH	See section 6.2.32.18
0x0012	modulusSizeL	See section 6.2.32.19
0x0013	modulusSizeH	See section 6.2.32.20
0x0014	exponentSizeL	See section 6.2.32.21
0x0015	exponentSizeH	See section 6.2.32.22
0x0016	lad_srevL	See section 6.2.32.23
0x0017	lad_srevH	See section 6.2.32.24
0x0018	reserved1L	See section 6.2.32.25
0x0019	reserved1H	See section 6.2.32.26
0x001A	lad_fw_entry_offsetL	See section 6.2.32.27
0x001B	lad_fw_entry_offsetH	See section 6.2.32.28
0x001C	reserved2L	See section 6.2.32.29
0x001D	reserved2H	See section 6.2.32.30
0x001E	lad_image_unique_idL	See section 6.2.32.31
0x001F	lad_image_unique_idH	See section 6.2.32.32
0x0020	lad_module_idL	See section 6.2.32.33
0x0021	lad_module_idH	See section 6.2.32.34
0x0022 + 1*n, n=0...31	Reserved	See section 6.2.32.35
0x0042 + 1*n, n=0...127	RSA Public Key	See section 6.2.32.36
0x00C2	RSA ExponentL	See section 6.2.32.37
0x00C3	RSA ExponentH	See section 6.2.32.38
0x00C4 + 1*n, n=0...127	Encrypted SHA256 Hash	See section 6.2.32.39
0x0144	Device Blank NVM Device ID	See section 6.2.32.40
0x0145	Max Module AreaL	See section 6.2.32.41
0x0146	Max Module AreaH	See section 6.2.32.42
0x0147	Current Module AreaL	See section 6.2.32.43
0x0148	Current Module AreaH	See section 6.2.32.44



Word Offset	Description	Reference
0x0149	Format Version + CRC	See section 6.2.32.45
0x014A	Code Revision	See section 6.2.32.46
0x014B	Reserved Spare Word	See section 6.2.32.47

6.2.32.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		

6.2.32.2 PCIe Analog Data - 0x0001

Raw data module length: variable



6.2.32.3 moduleTypeL - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	moduleType	0x6	

6.2.32.4 moduleTypeH - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	moduleTypeH		

6.2.32.5 headerLenL - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	headerLenL	0xA1	

6.2.32.6 headerLenH - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	headerLenH		

6.2.32.7 headerVersionL - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionL	0x00010000	

6.2.32.8 headerVersionH - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionH		

6.2.32.9 moduleIDL - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	moduleIDL	0x0	

6.2.32.10 moduleIDH - 0x0009

Bits	Field Name	Default NVM Value	Description
15	signMode	0x0	
14:0	moduleIDH		



6.2.32.11 moduleVendorL - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorL	0x00008086	

6.2.32.12 moduleVendorH - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorH		

6.2.32.13 dateL - 0x000C

0xMMDD

Bits	Field Name	Default NVM Value	Description
15:0	DateL	0x20130530	0xMMDD

6.2.32.14 dateH - 0x000D

0xYYYY

Bits	Field Name	Default NVM Value	Description
15:0	DateH		



6.2.32.15 sizeL - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	sizeL	0x00000800	

6.2.32.16 sizeH - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	sizeH		

6.2.32.17 keySizeL - 0x0010

Bits	Field Name	Default NVM Value	Description
15:0	keySizeL	0x40	

6.2.32.18 keySizeH - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	keySizeH		

6.2.32.19 modulusSizeL - 0x0012

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeL	0x40	

6.2.32.20 modulusSizeH - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeH		

6.2.32.21 exponentSizeL - 0x0014

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeL	0x1	

6.2.32.22 exponentSizeH - 0x0015

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeH		



6.2.32.23 lad_srevL - 0x0016

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevL	0x0	

6.2.32.24 lad_srevH - 0x0017

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevH		

6.2.32.25 reserved1L - 0x0018

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.32.26 reserved1H - 0x0019

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.32.27 lad_fw_entry_offsetL - 0x001A

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetL	0x14C	

6.2.32.28 lad_fw_entry_offsetH - 0x001B

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetH		

6.2.32.29 reserved2L - 0x001C

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.32.30 reserved2H - 0x001D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.



6.2.32.31 lad_image_unique_idL - 0x001E

Bits	Field Name	Default NVM Value	Description
15:0	lad_image_unique_idL	0x0	

6.2.32.32 lad_image_unique_idH - 0x001F

Bits	Field Name	Default NVM Value	Description
15:0	lad_image_unique_idH		

6.2.32.33 lad_module_idL - 0x0020

Bits	Field Name	Default NVM Value	Description
15:0	lad_module_idL	0x3	0x1 EMP image. 0x2 PE image. 0x3 PCIe analog. 0x4 PHY analog. 0x5 Option ROM.

6.2.32.34 lad_module_idH - 0x0021

Bits	Field Name	Default NVM Value	Description
15:0	lad_module_idH		

6.2.32.35 reserved[n] - 0x0022 + 1*n, n=0...31

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.32.36 RSA Public Key [n] - 0x0042 + 1*n, n=0...127

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.2.32.37 RSA ExponentL - 0x00C2

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentL	0x0	



6.2.32.38 RSA ExponentH - 0x00C3

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentH	0x0	

6.2.32.39 Encrypted SHA256 Hash[n] - 0x00C4 + 1*n, n=0...127

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.2.32.40 Device Blank NVM Device ID - 0x0144

Bits	Field Name	Default NVM Value	Description
15:0	Device Blank NVM Device ID	0x154B	

6.2.32.41 Max Module AreaL - 0x0145

Bits	Field Name	Default NVM Value	Description
15:0	Max Module AreaL	0x1000	

6.2.32.42 Max Module AreaH - 0x0146

Bits	Field Name	Default NVM Value	Description
15:0	Max Module AreaH	0x0000	

6.2.32.43 Current Module AreaL - 0x0147

Bits	Field Name	Default NVM Value	Description
15:0	Current Module AreaL	0x1000	

6.2.32.44 Current Module AreaH - 0x0148

Bits	Field Name	Default NVM Value	Description
15:0	Current Module AreaH	0x0000	



6.2.32.45 Format Version + CRC - 0x0149

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Format Version	0x2	
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: PCIe Analog -> Section Length End Section -> Word: PCIe Analog -> Reserved Spare Word

6.2.32.46 Code Revision - 0x014A

Bits	Field Name	Default NVM Value	Description
15:8	Major Revision	0x0	
7:0	Minor Revision	0x0	

6.2.32.47 Reserved Spare Word - 0x014B

Bits	Field Name	Default NVM Value	Description
15:0	Reserved Spare Word	0x0	

6.2.33 PHY Analog Section Summary Table

Configure the analog section of the SerDes PHY.

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.33.1
0x0001	PHY Analog Data	See section 6.2.33.2
0x0002	moduleTypeL	See section 6.2.33.3
0x0003	moduleTypeH	See section 6.2.33.4
0x0004	headerLenL	See section 6.2.33.5
0x0005	headerLenH	See section 6.2.33.6
0x0006	headerVersionL	See section 6.2.33.7
0x0007	headerVersionH	See section 6.2.33.8
0x0008	moduleIDL	See section 6.2.33.9
0x0009	moduleIDH	See section 6.2.33.10
0x000A	moduleVendorL	See section 6.2.33.11
0x000B	moduleVendorH	See section 6.2.33.12
0x000C	dateL	See section 6.2.33.13
0x000D	dateH	See section 6.2.33.14
0x000E	sizeL	See section 6.2.33.15
0x000F	sizeH	See section 6.2.33.16
0x0010	keySizeL	See section 6.2.33.17



Word Offset	Description	Reference
0x0011	keySizeH	See section 6.2.33.18
0x0012	modulusSizeL	See section 6.2.33.19
0x0013	modulusSizeH	See section 6.2.33.20
0x0014	exponentSizeL	See section 6.2.33.21
0x0015	exponentSizeH	See section 6.2.33.22
0x0016	lad_srevL	See section 6.2.33.23
0x0017	lad_srevH	See section 6.2.33.24
0x0018	reserved1L	See section 6.2.33.25
0x0019	reserved1H	See section 6.2.33.26
0x001A	lad_fw_entry_offsetL	See section 6.2.33.27
0x001B	lad_fw_entry_offsetH	See section 6.2.33.28
0x001C	reserved2L	See section 6.2.33.29
0x001D	reserved2H	See section 6.2.33.30
0x001E	lad_image_unique_idL	See section 6.2.33.31
0x001F	lad_image_unique_idH	See section 6.2.33.32
0x0020	lad_module_idL	See section 6.2.33.33
0x0021	lad_module_idH	See section 6.2.33.34
0x0022 + 1*n, n=0...31	Reserved	See section 6.2.33.35
0x0042 + 1*n, n=0...127	RSA Public Key	See section 6.2.33.36
0x00C2	RSA ExponentL	See section 6.2.33.37
0x00C3	RSA ExponentH	See section 6.2.33.38
0x00C4 + 1*n, n=0...127	Encrypted SHA256 Hash	See section 6.2.33.39
0x0144	Device Blank NVM Device ID	See section 6.2.33.40
0x0145	Max Module AreaL	See section 6.2.33.41
0x0146	Max Module AreaH	See section 6.2.33.42
0x0147	Current Module AreaL	See section 6.2.33.43
0x0148	Current Module AreaH	See section 6.2.33.44
0x0149	Format Version + CRC	See section 6.2.33.45
0x014A	Code Revision	See section 6.2.33.46
0x014B	Reserved Spare Word	See section 6.2.33.47

6.2.33.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		



6.2.33.2 PHY Analog Data - 0x0001

Raw data module length: variable



6.2.33.3 moduleTypeL - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	moduleType	0x6	

6.2.33.4 moduleTypeH - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	moduleTypeH		

6.2.33.5 headerLenL - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	headerLenL	0xA1	

6.2.33.6 headerLenH - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	headerLenH		

6.2.33.7 headerVersionL - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionL	0x00010000	

6.2.33.8 headerVersionH - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionH		

6.2.33.9 moduleIDL - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	moduleIDL	0x0	

6.2.33.10 moduleIDH - 0x0009

Bits	Field Name	Default NVM Value	Description
15	signMode	0x0	
14:0	moduleIDH		



6.2.33.11 moduleVendorL - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorL	0x00008086	

6.2.33.12 moduleVendorH - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorH		

6.2.33.13 dateL - 0x000C

0xMMDD

Bits	Field Name	Default NVM Value	Description
15:0	DateL	0x20130530	0xMMDD

6.2.33.14 dateH - 0x000D

0xYYYY

Bits	Field Name	Default NVM Value	Description
15:0	DateH		



6.2.33.15 sizeL - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	sizeL	0x00000800	

6.2.33.16 sizeH - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	sizeH		

6.2.33.17 keySizeL - 0x0010

Bits	Field Name	Default NVM Value	Description
15:0	keySizeL	0x40	

6.2.33.18 keySizeH - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	keySizeH		

6.2.33.19 modulusSizeL - 0x0012

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeL	0x40	

6.2.33.20 modulusSizeH - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeH		

6.2.33.21 exponentSizeL - 0x0014

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeL	0x1	

6.2.33.22 exponentSizeH - 0x0015

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeH		



6.2.33.23 lad_srevL - 0x0016

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevL	0x0	

6.2.33.24 lad_srevH - 0x0017

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevH		

6.2.33.25 reserved1L - 0x0018

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.33.26 reserved1H - 0x0019

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.33.27 lad_fw_entry_offsetL - 0x001A

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetL	0x14C	

6.2.33.28 lad_fw_entry_offsetH - 0x001B

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetH		

6.2.33.29 reserved2L - 0x001C

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.33.30 reserved2H - 0x001D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.



6.2.33.31 lad_image_unique_idL - 0x001E

Bits	Field Name	Default NVM Value	Description
15:0	lad_image_unique_idL	0x0	

6.2.33.32 lad_image_unique_idH - 0x001F

Bits	Field Name	Default NVM Value	Description
15:0	lad_image_unique_idH		

6.2.33.33 lad_module_idL - 0x0020

Bits	Field Name	Default NVM Value	Description
15:0	lad_module_idL	0x4	0x1 = EMP Image. 0x2 = PE Image. 0x3 = PCIe Analog. 0x4 = PHY Analog. 0x5 = Option ROM.

6.2.33.34 lad_module_idH - 0x0021

Bits	Field Name	Default NVM Value	Description
15:0	lad_module_idH		

6.2.33.35 reserved[n] - 0x0022 + 1*n, n=0...31

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.33.36 RSA Public Key [n] - 0x0042 + 1*n, n=0...127

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.2.33.37 RSA ExponentL - 0x00C2

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentL	0x0	



6.2.33.38 RSA ExponentH - 0x00C3

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentH	0x0	

6.2.33.39 Encrypted SHA256 Hash[n] - 0x00C4 + 1*n, n=0...127

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.2.33.40 Device Blank NVM Device ID - 0x0144

Bits	Field Name	Default NVM Value	Description
15:0	Device Blank NVM Device ID	0x154B	

6.2.33.41 Max Module AreaL - 0x0145

Bits	Field Name	Default NVM Value	Description
15:0	Max Module AreaL	0x1000	

6.2.33.42 Max Module AreaH - 0x0146

Bits	Field Name	Default NVM Value	Description
15:0	Max Module AreaH	0x0000	

6.2.33.43 Current Module AreaL - 0x0147

Bits	Field Name	Default NVM Value	Description
15:0	Current Module AreaL	0x1000	

6.2.33.44 Current Module AreaH - 0x0148

Bits	Field Name	Default NVM Value	Description
15:0	Current Module AreaH	0x0000	



6.2.33.45 Format Version + CRC - 0x0149

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Format Version	0x2	
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: PHY Analog -> Section Length End Section -> Word: PHY Analog -> Reserved Spare Word

6.2.33.46 Code Revision - 0x014A

Bits	Field Name	Default NVM Value	Description
15:8	Major Revision	0x0	
7:0	Minor Revision	0x0	

6.2.33.47 Reserved Spare Word - 0x014B

Bits	Field Name	Default NVM Value	Description
15:0	Reserved Spare Word	0x0	

6.2.34 DL Scripts Section Summary Table

6.2.35 EMP Global Module Section Summary Table

This section contains two subsections:

1. List of Qualified Modules — Parameter list of up to 16 modules. Per module list holds OUI, revision and version numbers.
2. 10 GbE Link Establishment State Machine (LESM) settings.

Word Offset	Description	Reference
0x13000	Section Length	See section 6.2.35.1
0x13001	Number of Qualified Modules	See section 6.2.35.2
0x13002 + 12*n, n=0...15	Module OUI Bytes 0-1	See section 6.2.35.3
0x13003 + 12*n, n=0...15	Module OUI Byte 2	See section 6.2.35.4
0x13004 + 12*n, n=0...15	Vendor Part Number Bytes 0-1	See section 6.2.35.5
0x13005 + 12*n, n=0...15	Vendor Part Number Bytes 2-3	See section 6.2.35.6
0x13006 + 12*n, n=0...15	Vendor Part Number Bytes 4-5	See section 6.2.35.7
0x13007 + 12*n, n=0...15	Vendor Part Number Bytes 6-7	See section 6.2.35.8



Word Offset	Description	Reference
0x13008 + 12*n, n=0...15	Vendor Part Number Bytes 8-9	See section 6.2.35.9
0x13009 + 12*n, n=0...15	Vendor Part Number Bytes 10-11	See section 6.2.35.10
0x1300A + 12*n, n=0...15	Vendor Part Number Bytes 12-13	See section 6.2.35.11
0x1300B + 12*n, n=0...15	Vendor Part Number Bytes 14-15	See section 6.2.35.12
0x1300C + 12*n, n=0...15	Module Revision Number Bytes 0-1	See section 6.2.35.13
0x1300D + 12*n, n=0...15	Module Revision Number Bytes 2-3	See section 6.2.35.14
0x130C2	LESM Global Configurations	See section 6.2.35.15
0x130C3	LESM AN73 + PD State Configurations	See section 6.2.35.16
0x130C4	Reserved	See section 6.2.35.17
0x130C5	Reserved	See section 6.2.35.18
0x130C6	CRC8	See section 6.2.35.19

6.2.35.1 Section Length - 0x13000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.

6.2.35.2 Number of Qualified Modules - 0x13001

Number of valid entries, 0 through 15, in the qualified modules' list.

Bits	Field Name	Default NVM Value	Description
15:0	Number of Qualified Modules	0x10	Note: This field is preserved by the Intel NVM update tool.



6.2.35.3 Module OUI Bytes 0-1[n] - 0x13002 + 12*n, n=0...15

OUI of qualified external modules.

SFP+: Address 0xA0, Byte 3-5.

Bits	Field Name	Default NVM Value	Description
15:8	Byte 1	0x1B	Note: This field is preserved by the Intel NVM update tool.
7:0	Byte 0	0x00	Note: This field is preserved by the Intel NVM update tool.

6.2.35.4 Module OUI Byte 2[n] - 0x13003 + 12*n, n=0...15

OUI of qualified external modules.

SFP+: Address 0xA0, Byte 3-5.

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	Reserved.
7:0	Byte 0	0x21	Note: This field is preserved by the Intel NVM update tool.

6.2.35.5 Vendor Part Number Bytes 0-1[n] - 0x13004 + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte1	0x0	
7:0	Byte0	0x0	

6.2.35.6 Vendor Part Number Bytes 2-3[n] - 0x13005 + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte1	0x0	
7:0	Byte0	0x0	



6.2.35.7 Vendor Part Number Bytes 4-5[n] - 0x13006 + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte1	0x0	
7:0	Byte0	0x0	

6.2.35.8 Vendor Part Number Bytes 6-7[n] - 0x13007 + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte1	0x0	
7:0	Byte0	0x0	

6.2.35.9 Vendor Part Number Bytes 8-9[n] - 0x13008 + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte1	0x0	
7:0	Byte0	0x0	

6.2.35.10 Vendor Part Number Bytes 10-11[n] - 0x13009 + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte1	0x0	
7:0	Byte0	0x0	



6.2.35.11 Vendor Part Number Bytes 12-13[n] - 0x1300A + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte1	0x0	
7:0	Byte0	0x0	

6.2.35.12 Vendor Part Number Bytes 14-15[n] - 0x1300B + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte1	0x0	
7:0	Byte0	0x0	

6.2.35.13 Module Revision Number Bytes 0-1[n] - 0x1300C + 12*n, n=0...15

Vendor part number of qualified external modules.

SFP+: Address 0xA0 Bytes 55:40.

Bits	Field Name	Default NVM Value	Description
15:8	Byte 1	0x0	
7:0	Byte 0	0x0	

6.2.35.14 Module Revision Number Bytes 2-3[n] - 0x1300D + 12*n, n=0...15

Revision number of qualified external modules.

SFP+: Address 0xA0 Bytes 59:56.

Bits	Field Name	Default NVM Value	Description
15:8	Byte 1	0x0	
7:0	Byte 0	0x0	



6.2.35.15 LESM Global Configurations - 0x130C2

Bits	Field Name	Default NVM Value	Description
15:6	Reserved	0x0	Reserved.
5:1	10G Only Counter	0x3	Number of attempts to make at 10 GbE before attempting to establish the link at 1 GbE.
0	LESM Enable	0x0	1b = LESM is enabled. 0b = LESM is disabled. Valid values are: 0x0 = Disabled. 0x1 = Enabled.

6.2.35.16 LESM AN73 + PD State Configurations - 0x130C3

Bits	Field Name	Default NVM Value	Description
15:1	Timeout	0x32	Number of seconds to stay in this state before moving on to the next. Granularity is in 0.1 seconds.
0	State Enable	0x1	1b = State is enabled. Note: This state must always be enabled when the LESM feature is enabled. Valid values are: 0x0 = Disabled. 0x1 = Enabled.

6.2.35.17 Reserved - 0x130C4

Bits	Field Name	Default NVM Value	Description
15:1	Reserved	0x32	Reserved.
0	Reserved	0x1	Reserved.

6.2.35.18 Reserved - 0x130C5

Bits	Field Name	Default NVM Value	Description
15:1	Reserved	0x32	Reserved.
0	Reserved	0x1	Reserved.

6.2.35.19 CRC8 - 0x130C6

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: EMP Global Module -> Section Length End Section -> Word: EMP Global Module -> Reserved.



6.2.36 Manageability Module Header Section Summary Table

This section contains parameters related to the manageability functionality such as connection type and others, It also points to subsections configuring the filters and the sideband interfaces.

Word Offset	Description	Reference
0x14000	Section Length	See section 6.2.36.1
0x14001	Common Manageability Parameters	See section 6.2.36.2
0x14002	Common Manageability Parameters 2	See section 6.2.36.3
0x14003	Pass Through LAN 0 Configuration Pointer	See section 6.2.36.4
0x14004	Pass Through LAN 1 Configuration Pointer	See section 6.2.36.5
0x14005	Pass Through LAN 2 Configuration Pointer	See section 6.2.36.6
0x14006	Pass Through LAN 3 Configuration Pointer	See section 6.2.36.7
0x14007	Sideband Configuration Pointer	See section 6.2.36.8
0x14008	Flexible TCO Filter Configuration Pointer	See section 6.2.36.9
0x14009	Traffic Types Parameters	See section 6.2.36.10
0x1400A	OEM Structure Pointer	See section 6.2.36.11
0x1400B	Thermal Sensor Setting Pointer	See section 6.2.36.12
0x1400C	CRC8	See section 6.2.36.13

6.2.36.1 Section Length - 0x14000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.



6.2.36.2 Common Manageability Parameters - 0x14001

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	Reserved.
10:8	Manageability Pass Through Mode	0x2	0x0 = None. 0x2 = PT mode.
5	Reserved	0x0	Reserved.
4	Force TCO Reset Disable	0x1	If cleared, enables the MC to do a global reset of the device using the Force TCO SMBus or NC-SI commands. 0x0 Enable Force TCO reset. 0x1 Disable Force TCO reset.
3	Reserved	0x0	0b = Disable. 1b = Enable.
2	OS2BMC Capable	0x1	0b = Disable. 1b = Enable. Valid values are: 0x0 = Disabled. 0x1 = Enabled.
1:0	Reserved	0x0	Reserved.

6.2.36.3 Common Manageability Parameters 2 - 0x14002

Bits	Field Name	Default NVM Value	Description
15:12	Reserved		Reserved.
11	Multi-Drop NC-SI	0x1	0x0 = Point-to-point. 0x1 = Multi-drop.
10	Reserved		Reserved.
9	EMP_LINK_ON	0x1	Copy of the PRTPM_GC.EMP_LINK_ON bit for EMP use. 0x0 = Disable. 0x1 = Enable.
8:7	Reserved	0x0	Reserved.
6	PLDM over MCTP Support	0x0	0x0 = Disabled. 0x1 = Enabled.
5	OEM over MCTP Support	0x0	0x0 = Disabled. 0x1 = Enabled.
4	NC-SI over MCTP Support	0x0	0x0 = Disabled. 0x1 = Enabled.
3	LAN3_FTCO_ISOL_DIS	0x1	0x0 = Disabled. 0x1 = Enabled.
2	LAN2_FTCO_ISOL_DIS	0x1	0x0 = Disabled. 0x1 = Enabled.
1	LAN1_FTCO_ISOL_DIS	0x1	0x0 = Disabled. 0x1 = Enabled.
0	LAN0_FTCO_ISOL_DIS	0x1	0x0 = Disabled. 0x1 = Enabled.



6.2.36.4 Pass Through LAN 0 Configuration Pointer - 0x14003

Bits	Field Name	Default NVM Value	Description
15:0	Pass Through LAN 0 Configuration Pointer	0xFFFF	

The Pass Through LAN 0 Configuration Pointer points to the Pass Through Control Words Structure 0 section. For Pass Through Control Words Structure 0 inner structure detail, see [Section 6.2.37](#).

6.2.36.5 Pass Through LAN 1 Configuration Pointer - 0x14004

Bits	Field Name	Default NVM Value	Description
15:0	Pass Through LAN 1 Configuration Pointer	0xFFFF	

The Pass Through LAN 1 Configuration Pointer points to the Pass Through Control Words Structure 1 section. For Pass Through Control Words Structure 1 inner structure detail, see [Section 6.2.37](#).

6.2.36.6 Pass Through LAN 2 Configuration Pointer - 0x14005

Bits	Field Name	Default NVM Value	Description
15:0	Pass Through LAN 2 Configuration Pointer	0xFFFF	

The Pass Through LAN 2 Configuration Pointer points to the Pass Through Control Words Structure2 section. For Pass Through Control Words Structure2 inner structure detail, see [Section 6.2.37](#).

6.2.36.7 Pass Through LAN 3 Configuration Pointer - 0x14006

Bits	Field Name	Default NVM Value	Description
15:0	Pass Through LAN 3 Configuration Pointer	0xFFFF	

The Pass Through LAN 3 Configuration Pointer points to the Pass Through Control Words Structure3 section. For Pass Through Control Words Structure3 inner structure detail, see [Section 6.2.37](#).

6.2.36.8 Sideband Configuration Pointer - 0x14007

This module is long by 28 bytes and must be mapped in the first valid 4 KB sector of the Flash.

Bits	Field Name	Default NVM Value	Description
15:0	Sideband Configuration Pointer	0xFFFF	

The Sideband Configuration Pointer points to the Sideband Configuration Structure section. For Sideband Configuration Structure inner structure detail, see [Section 6.2.39](#).



6.2.36.9 Flexible TCO Filter Configuration Pointer - 0x14008

This section loads all flexible filters. The control and mask and filter data are repeatable as the number of filters. Section length in offset 0 is for all filters.

Bits	Field Name	Default NVM Value	Description
15:0	Flexible TCO Filter Configuration Pointer	0xFFFF	

The Flexible TCO Filter Configuration Pointer points to the Flexible TCO Filter Configuration Structure section. For Flexible TCO Filter Configuration Structure inner structure detail, see [Section 6.2.38](#).



6.2.36.10 Traffic Types Parameters - 0x14009

Bits	Field Name	Default NVM Value	Description
15:14	Reserved	0x0	Reserved.
13:12	Port 3 Traffic Types	0x1	Defines which type of traffic can flow to and from primary BMC connection on port 3. The traffic types defined by this field are enabled by the <i>Manageability Mode</i> field and the <i>OS2BMC Capable</i> bit in the Common Manageability Parameters 1 NVM word. 0x0 = Reserved. 0x1 = Network to BMC traffic only. 0x2 = OS2BMC traffic only. 0x3 = Both Network to BMC traffic and OS2BMC traffic.
11:10	Reserved	0x0	Reserved.
9:8	Port 2 Traffic Types	0x1	Defines which type of traffic can flow to and from primary BMC connection on port 2. 0x0 = Reserved. 0x1 = Network to BMC traffic only. 0x2 = OS2BMC traffic only. 0x3 = Both Network to BMC traffic and OS2BMC traffic.
7:6	Reserved	0x0	Reserved.
5:4	Port 1 Traffic Types	0x1	Defines which type of traffic can flow to and from primary BMC connection on port 1. 0x0 = Reserved. 0x1 = Network to BMC traffic only. 0x2 = OS2BMC traffic only. 0x3 = Both Network to BMC traffic and OS2BMC traffic.
3:2	Reserved	0x0	Reserved.
1:0	Port 0 Traffic Types	0x1	Defines which type of traffic can flow to and from primary BMC connection on port 0. 0x0 = Reserved. 0x1 = Network to BMC traffic only. 0x2 = OS2BMC traffic only. 0x3 = Both Network to BMC traffic and OS2BMC traffic.

6.2.36.11 OEM Structure Pointer - 0x1400A

Bits	Field Name	Default NVM Value	Description
15:0	OEM Structure Pointer	0xFFFF	

The OEM Structure Pointer points to the OEM Section. For OEM Section inner structure detail, see [Section 6.2.40](#).

6.2.36.12 Thermal Sensor Setting Pointer - 0x1400B

Bits	Field Name	Default NVM Value	Description
15:0	Thermal Sensor Setting Pointer	0xFFFF	

The Thermal Sensor Setting Pointer points to the Thermal Sensor Setting section. For Thermal Sensor Setting inner structure detail, see [Section 6.2.40](#).



6.2.36.13 CRC8 - 0x1400C

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: Manageability Module Header -> Section Length End Section -> Word: Manageability Module Header -> Thermal Sensor Setting Pointer

6.2.37 Pass Through Control Words Structure 0 Section Summary Table

This section contains the initial setting of the pass through filters for port 0. This section is used only in SMBus legacy pass through mode.

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.37.1
0x0001 + 2*n, n=0...3	LAN IPv4 Address (LSB) MIPAF0	See section 6.2.37.2
0x0002 + 2*n, n=0...3	LAN IPv4 Address (MSB) MIPAF0	See section 6.2.37.3
0x0009 + 2*n, n=0...15	LAN Flexible Filter Port	See section 6.2.37.4
0x000A + 2*n, n=0...15	LAN Flexible Filter Port - Modifiers	See section 6.2.37.5
0x0029 + 1*n, n=0...7	LAN VLAN Filter	See section 6.2.37.6
0x0031	LAN MANC Value LSB	See section 6.2.37.7
0x0032	LAN MANC Value MSB	See section 6.2.37.8
0x0033	Receive Enable 1 (LRXEN1)	See section 6.2.37.9
0x0034	Receive Enable 2 (LRXEN2)	See section 6.2.37.10
0x0035	LAN MNGONLY LSB	See section 6.2.37.11
0x0036	LAN MNGONLY MSB	See section 6.2.37.12
0x0037 + 4*n, n=0...6	Manageability Decision Filters LSB	See section 6.2.37.13
0x0038 + 4*n, n=0...6	Manageability Decision Filters MSB	See section 6.2.37.14
0x0039 + 4*n, n=0...6	Manageability Extended Decision Filters LSB	See section 6.2.37.15
0x003A + 4*n, n=0...6	Manageability Extended Decision Filters MSB	See section 6.2.37.16
0x0053 + 2*n, n=0...3	Manageability Ethertype filter - METF - LSB	See section 6.2.37.17
0x0054 + 2*n, n=0...3	Manageability Ethertype filter METF - MSB	See section 6.2.37.18
0x005B	ARP Response IPv4 Address (LSB)	See section 6.2.37.19
0x005C	ARP Response IPv4 Address (MSB)	See section 6.2.37.20



Word Offset	Description	Reference
0x005D + 8*n, n=0...3	IPv6 Address Bytes 0-1	See section 6.2.37.21
0x005E + 8*n, n=0...3	IPv6 Address Bytes 2-3	See section 6.2.37.22
0x005F + 8*n, n=0...3	IPv6 Address Bytes 4-5	See section 6.2.37.23
0x0060 + 8*n, n=0...3	IPv6 Address Bytes 6-7	See section 6.2.37.24
0x0061 + 8*n, n=0...3	IPv6 Address Bytes 8-9	See section 6.2.37.25
0x0062 + 8*n, n=0...3	IPv6 Address Bytes 10-11	See section 6.2.37.26
0x0063 + 8*n, n=0...3	IPv6 Address Bytes 12-13	See section 6.2.37.27
0x0064 + 8*n, n=0...3	IPv6 Address Bytes 14-15	See section 6.2.37.28
0x007D	Manageability Special Modifiers - LSB	See section 6.2.37.29
0x007E	Manageability Special Modifiers - MSB	See section 6.2.37.30
0x007F	CRC8	See section 6.2.37.31

6.2.37.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.



6.2.37.2 LAN IPv4 Address (LSB) MIPAF0[n] = 0x0001 + 2*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:8	IPv4 Address, Byte 1	0xFF	Note: This field is preserved by the Intel NVM update tool.
7:0	IPv4 Address, Byte 0	0xFF	Note: This field is preserved by the Intel NVM update tool.

6.2.37.3 LAN IPv4 Address (MSB) MIPAF0[n] - 0x0002 + 2*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:8	IPv4 Address, Byte 3	0xFF	Note: This field is preserved by the Intel NVM update tool.
7:0	IPv4 Address, Byte 2	0xFF	Note: This field is preserved by the Intel NVM update tool.

6.2.37.4 LAN Flexible Filter Port[n] - 0x0009 + 2*n, n=0...15

Bits	Field Name	Default NVM Value	Description
15:0	LAN UDP Flexible Filter Port0	0xFFFF	Note: This field is preserved by the Intel NVM update tool.

6.2.37.5 LAN Flexible Filter Port - Modifiers[n] - 0x000A + 2*n, n=0...15

Bits	Field Name	Default NVM Value	Description
15:3	Reserved		Reserved.
2	Source Destination	0x0	0x0 = Destination. 0x1 = Source. Note: This field is preserved by the Intel NVM update tool.
1	Accept TCP	0x0	If set, filter match if packet is TCP. 0x0 = Filter. 0x1 = Accept. Note: This field is preserved by the Intel NVM update tool.
0	Accept UDP	0x0	If set, filter match if packet is UDP. 0x0 = Filter. 0x1 = Accept. Note: This field is preserved by the Intel NVM update tool.

6.2.37.6 LAN VLAN Filter [n] - 0x0029 + 1*n, n=0...7

Bits	Field Name	Default NVM Value	Description
15:12	Reserved	0x0	Reserved.
11:0	VLAN Filter 0 Value	0xFFF	Note: This field is preserved by the Intel NVM update tool.

6.2.37.7 LAN MANC Value LSB - 0x0031

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.



6.2.37.8 LAN MANC Value MSB - 0x0032

Bits	Field Name	Default NVM Value	Description
15:11	Reserved	0x0	Reserved.
10	NET_TYPE	0x0	Note: This field is preserved by the Intel NVM update tool.
9	FIXED_NET_TYPE	0x0	Note: This field is preserved by the Intel NVM update tool.
8	Reserved	0x0	Reserved.
7	EN_XSUM_FILTER	0x0	Should be used to set the RBCR.EN_XSUM_FILTERn bit (n is the port number). Note: This field is preserved by the Intel NVM update tool.
6:0	Reserved	0x0	Reserved.

6.2.37.9 Receive Enable 1 - (LRXEN1) - 0x0033

Bits	Field Name	Default NVM Value	Description
15:9	Receive Enable byte 12	0x00	BMC SMBus slave address. Note: This field is preserved by the Intel NVM update tool.
8	Reserved	0x0	Reserved.
7	Enable BMC Dedicated	0x0	When set, the BMC receives traffic sent to the MAC address defined in MMAH/MMAL[3]. Firmware configures MDEF, MDEF_EXT (7) to a dedicated MAC configured into MMAL/H (3) to implement this mode. Note: This field is preserved by the Intel NVM update tool.
6	Reserved	0x1	Reserved.
5:4	Notification method	0x0	0x0 = SMBus alert. 0x1 = Asynchronous notify. 0x2 = Direct receive. 0x3 = Reserved. Note: This field is preserved by the Intel NVM update tool.
3	Enable ARP Response	0x0	When set, firmware offloads ARP handling sent to the IP address defined in the ARP Response IPv4 Address NVM words. Firmware uses MDEF, MDEF_EXT (6) and MIPAF4[3] to implement this mode. Note: This field is preserved by the Intel NVM update tool.
2	Enable Status Reporting	0x0	Note: This field is preserved by the Intel NVM update tool.
1	Enable Receive All	0x0	Note: This field is preserved by the Intel NVM update tool.
0	Enable Receive TCO	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.37.10 Receive Enable 2 (LRXEN2) - 0x0034

Bits	Field Name	Default NVM Value	Description
15:8	Receive Enable byte 14	0x00	Alert Value. Note: This field is preserved by the Intel NVM update tool.
7:0	Receive Enable byte 13	0x00	Interface Data. Note: This field is preserved by the Intel NVM update tool.



6.2.37.11 LAN MNGONLY LSB - 0x0035

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	Reserved.
7:0	Exclusive to MNG	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.37.12 LAN MNGONLY MSB - 0x0036

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.37.13 Manageability Decision Filters LSB[n] - 0x0037 + 4*n, n=0...6

Bits	Field Name	Default NVM Value	Description
15:0	MDEF0_L	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.37.14 Manageability Decision Filters MSB[n] - 0x0038 + 4*n, n=0...6

Bits	Field Name	Default NVM Value	Description
15:0	MDEF0_M	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.37.15 Manageability Extended Decision Filters LSB[n] - 0x0039 + 4*n, n=0...6

Bits	Field Name	Default NVM Value	Description
15:0	MDEFEXT0_L	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.37.16 Manageability Extended Decision Filters MSB[n] - 0x003A + 4*n, n=0...6

Bits	Field Name	Default NVM Value	Description
15:0	MDEFEXT0_M	0x0	Note: This field is preserved by the Intel NVM update tool.

6.2.37.17 Manageability Ethertype Filter - METF - LSB[n] -0x0053 + 2*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	METF0_L	0x0000	Loaded to 16 LS bits of METF[0] register. Note: This field is preserved by the Intel NVM update tool.



6.2.37.18 Manageability Ethertype Filter METF - MSB[n] - 0x0054 + 2*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	METF0_M	0x0000	Loaded to 16 MS bits of METF[0] register (reserved bits in the METF registers should be set in the NVM to the register default values) Note: This field is preserved by the Intel NVM update tool.

6.2.37.19 ARP Response IPv4 Address (LSB) - 0x005B

Bits	Field Name	Default NVM Value	Description
15:8	Byte 1	0x00	Firmware use. Note: This field is preserved by the Intel NVM update tool.
7:0	Byte 0	0x00	Firmware use. Note: This field is preserved by the Intel NVM update tool.

6.2.37.20 ARP Response IPv4 Address (MSB) - 0x005C

Bits	Field Name	Default NVM Value	Description
15:8	Byte 3	0x00	Firmware use. Note: This field is preserved by the Intel NVM update tool.
7:0	Byte 2	0x00	Firmware use. Note: This field is preserved by the Intel NVM update tool.

6.2.37.21 IPv6 Address Bytes 0-1[n] - 0x005D + 8*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	IPv6 Address Bytes 0-1	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.37.22 IPv6 Address Bytes 2-3[n] - 0x005E + 8*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	IPv6 Address Bytes 2-3	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.37.23 IPv6 Address Bytes 4-5[n] - 0x005F + 8*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	IPv6 Address Bytes 4-5	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.37.24 IPv6 Address Bytes 6-7[n] - 0x0060 + 8*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	IPv6 Address Bytes 6-7	0x0000	Note: This field is preserved by the Intel NVM update tool.

**6.2.37.25 IPv6 Address Bytes 8-9[n] - 0x0061 + 8*n, n=0...3**

Bits	Field Name	Default NVM Value	Description
15:0	IPv6 Address Bytes 8-9	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.37.26 IPv6 Address Bytes 10-11[n] - 0x0062 + 8*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	IPv6 Address Bytes 10-11	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.37.27 IPv6 Address Bytes 12-13[n] - 0x0063 + 8*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	IPv6 Address Bytes 12-13	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.37.28 IPv6 Address Bytes 14-15[n] - 0x0064 + 8*n, n=0...3

Bits	Field Name	Default NVM Value	Description
15:0	IPv6 Address Bytes 14-15	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.37.29 Manageability Special Modifiers - LSB - 0x007D

Bits	Field Name	Default NVM Value	Description
15:0	MSFM_L	0x000F	Note: This field is preserved by the Intel NVM update tool.

6.2.37.30 Manageability Special Modifiers - MSB - 0x007E

Bits	Field Name	Default NVM Value	Description
15:0	MSFM_M	0x0000	Note: This field is preserved by the Intel NVM update tool.

6.2.37.31 CRC8 - 0x007F

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: Pass Through Control Words Structure 0 -> Section Length End Section -> Word: Pass Through Control Words Structure 0 -> Manageability Special Modifiers - MSB



6.2.38 Flexible TCO Filter Configuration Structure Section Summary Table

This section contains the setting of the manageability flex filters for all ports. Its format is described in the Manageability section (Flexible TCO Filter Configuration).

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.38.1
0x0001	Flexible Filter Data	See section 6.2.38.2
0x0002	CRC8	See section 6.2.38.3

6.2.38.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in: 2 bytes unit - 1 First Section -> Word: Flexible TCO Filter Configuration Structure -> Section Length Last Section -> Word: Flexible TCO Filter Configuration Structure -> CRC8

6.2.38.2 Flexible Filter Data - 0x0001

Raw data module length: variable

Flexible Filter Data.

Note: This word is preserved by Intel NVM update tool.



6.2.38.3 CRC8 - 0x0002

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: Flexible TCO Filter Configuration Structure -> Section Length End Section -> Word: Flexible TCO Filter Configuration Structure -> Flexible Filter Data

6.2.39 Sideband Configuration Structure Section Summary Table

This section describes the setting of the different pass through interfaces (SMBus, NC-SI and MCTP).

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.39.1
0x0001	SMBus Maximum Fragment Size	See section 6.2.39.2
0x0002	SMBus Notification Timeout and Flags	See section 6.2.39.3
0x0003	NC-SI Configuration 1	See section 6.2.39.4
0x0004	NC-SI Configuration 2	See section 6.2.39.5
0x0005	NC-SI Flow Control XOFF	See section 6.2.39.6
0x0006	NC-SI Flow Control XON	See section 6.2.39.7
0x0007	NC-SI Hardware Arbitration Configuration	See section 6.2.39.8
0x000D	OEM IANA	See section 6.2.39.9
0x000E	NC-SI over MCTP Message Types	See section 6.2.39.10
0x000F	NC-SI over MCTP Configuration	See section 6.2.39.11
0x0010	MCTP Rate Limiter Config 1	See section 6.2.39.12
0x0011	MCTP Rate Limiter Config 2	See section 6.2.39.13
0x0012	NC-SI Channel to Port Mapping	See section 6.2.39.14
0x0013	CRC8	See section 6.2.39.15

6.2.39.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.



6.2.39.2 SMBus Maximum Fragment Size - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Fragment Size	0x0020	SMBus Maximum Fragment Size (bytes). Supported range is between 32 and 240 bytes. Note: In MCTP mode, this value should be set to 0x45 (64 bytes payload + 5 bytes of MCTP header).

6.2.39.3 SMBus Notification Timeout and Flags - 0x0002

Bits	Field Name	Default NVM Value	Description
15:8	SMBus Notification Timeout (ms)	0xFF	
7:6	SMBus Connection Speed	0x0	0x0 = Standard SMBus connection. 0x1 = 400 Kb/ps fast I ² C. 0x2 = 1 Mb/s fast + I ² C. 0x3 = Reserved.
5	SMBus Block Read Command	0x0	0x0 = Block read command is 0xC0. 0x1 = Block read command is 0xD0.
4	Reserved	0x0	Reserved.
3	Enable Fairness Arbitration	0x1	MCTP over SMBus feature. 0b = Disable fairness arbitration. 1b = Enable fairness arbitration. Valid values are: 0x0 = Disabled. 0x1 = Enabled.
2	Disable SMBus ARP Functionality	0x0	0x0 = SMBus ARP enabled. 0x1 = SMBus ARP disabled.
1	SMBus ARP PEC	0x1	SMBus ARP PEC. Should be set in MCTP modes. 0x0 = Disable PEC - Disable SMBus ARP PEC. 0x1 = Enable PEC - Enable SMBus ARP PEC.
0	SMBus Transaction PEC	0x0	SMBus Transactions PEC Should be set in MCTP modes. 0x0 = Disable PEC. If this bit is cleared, PEC is not added to master write or slave read transactions, a slave write transaction with PEC is dropped. 0x1 = Enable PEC. If this bit is set, PEC is added for master SMBus write transactions. A PEC is added to slave read transactions and can be received in slave write transaction.

6.2.39.4 NC-SI Configuration 1 - 0x0003

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	Reserved.
14	Flow Control	0x0	0x0 = NC-SI flow control disable. 0x1 = NC-SI flow control enable.
13:10	NC-SI Version	0x1	Supported NC-SI version. 0x0 = NC-SI 1.0.1. 0x1 = NC-SI 1.1.
9	NC-SI HW Arbitration Enable	0x0	0x0 = Not supported. 0x1 = Supported.



Bits	Field Name	Default NVM Value	Description
7:5	Package ID	0x0	Meaningful only when bit 15 of NC-SI Configuration 2 word (offset 0x07) is cleared.
4:1	Reserved	0x0	Reserved - must be zero.



6.2.39.5 NC-SI Configuration 2 - 0x0004

Bits	Field Name	Default NVM Value	Description
15	Read NC-SI Package ID from SDP	0x0	Read NC-SI Package ID From. 0b = NVM, NC-SI Configuration 1 word bits 7:5 (offset 0x06). 1b = The SDP defined in the <i>PackageID</i> SDP field defines the package ID. In this case, the package ID can be 0 or 2 according to {0,SDP value,0}. Valid values are: 0x0 = PackageID from NVM. 0x1 = PackageID from SDP.
14:10	PackageID SDP	0x0	Defines the SDP from which the NC-SI package ID is taken.
9:8	Reserved	0x000	Reserved.
7:4	Reserved	0x1	Reserved.
3:0	Max XOFF Renewal	0x3	NC-SI Flow Control Maximum XOFF Renewal (# of XOFF renewals allowed). 0x0 = Disabled. Unlimited number of XOFF frames can be sent. 0x1 = Up to 2 consecutive XOFFs frames can be sent by X710-TM4. 0x2 = Up to 3 consecutive XOFFs frames can be sent by X710-TM4. ... 0xF = Up to 16 consecutive XOFFs frames can be sent by X710-TM4.

6.2.39.6 NC-SI Flow Control XOFF - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	XOFF Threshold	0x12B8	Tx buffer watermark for sending a XOFF NC-SI flow control packet in bytes. The XOFF threshold value refers to the occupied space in the buffer. Notes: Field relevant for NC-SI operation mode only. To support a maximum packet size of 1.5 KB, the value programmed assuming a Tx buffer size of 8 KB, the value of the field should be 0x12B8 (2,744 bytes).

6.2.39.7 NC-SI Flow Control XON - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	XON Threshold	0x1348	Tx buffer water mark for sending a XON NC-SI flow control packet in bytes. The XON threshold value refers to the available space in the Tx buffer. Notes: Field relevant for NC-SI operation mode only. To support maximum packet size of 1.5 KB, the value programmed should be a positive value that equals: buffer size - XOFF threshold + 1536 bytes. Assuming a Tx buffer size is 8 KB and the XOFF threshold is 4792 bytes, the value of the field should be 0x1348 (4,936 bytes).

6.2.39.8 NC-SI Hardware Arbitration Configuration - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	TOKEN Timeout	0xA000	NC-SI Hardware Arbitration Token Timeout (in NC-SI REF_CLK cycles - 20 ns). Setting this value to 0x0 disables the timeout mechanism.



6.2.39.9 OEM IANA - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	OEM IANA	0x0	If not 0x0 and not 0x157, the device accepts additional OEM commands with this IANA number. These commands are accepted only if the OEM NVM structure pointer is valid. The set of commands accepted depends on the IANA value in this field. The regular Intel OEM commands are accepted only with IANA 0x157.

6.2.39.10 NC-SI over MCTP Message Types - 0x000E

Bits	Field Name	Default NVM Value	Description
15:8	NC-SI Control Message Type	0x2	Defines the MCTP message type used to identify NC-SI control packets.
7:0	NC-SI Pass Through Message type	0x3	Defines the MCTP message type used to identify NC-SI pass through packets.

6.2.39.11 NC-SI Over MCTP Configuration - 0x000F

Bits	Field Name	Default NVM Value	Description
15:7	Reserved	0x0	Reserved.
6	Simplified MCTP	0x0	If set, only <i>SOM</i> and <i>EOM</i> bits are used for the reassembly process. Relevant only in SMBus mode.
5	Disable ACLs	0x0	If set, the ACLs on the PCIe VDMs are disabled.
4:1	Reserved	0x0	Reserved.

6.2.39.12 MCTP Rate Limiter Config 1 - 0x0010

MCTP rate limiter configuration for first channel.

Bits	Field Name	Default NVM Value	Description
15:0	MCTP Rate	0x61A8	Defines the number of cycles between accesses of the MCTP send client to the memory arbiter. Current value assumes a clock of 195 MHz and a bus width of 128 bit. This value provides a bit rate of 1 Mb/s.

6.2.39.13 MCTP Rate Limiter Config 2 - 0x0011

MCTP rate limiter configuration for first channel.

Bits	Field Name	Default NVM Value	Description
15	Decision Point	0x0	Defines if, when credits are available, a full MCTP message is sent or a single VDM is sent. 0x0 = Per VDM. 0x1 = Per packet.
14:0	MCTP Max Credits	0x5	Defines the maximum number of 16 bytes credit that can be accumulated. These credits include the VDM header line (one line for each 64 byte VDM).



6.2.39.14 NC-SI Channel to Port Mapping - 0x0012

Defines the mapping of NC-SI channels to physical ports.

Note: The list of channel IDs must starts at zero and be consecutive.

Bits	Field Name	Default NVM Value	Description
15	Table Valid	0x0	0x0 = Table invalid use default algorithm described in the Channel ID mapping section. 0x1 = Table valid uses the mapping defined in this word.
14:13	Port #3 Channel ID	0x0	
12	Port #3 Channel Enable	0x1	0x0 = Disabled. 0x1 = Enabled.
11	Reserved	0x0	Reserved.
10:9	Port #2 Channel ID	0x0	
8	Port #2 Channel Enable	0x1	0x0 = Disabled. 0x1 = Enabled.
7	Reserved	0x0	Reserved.
6:5	Port #1 Channel ID	0x0	
4	Port #1 Channel Enable	0x1	0x0 = Disabled. 0x1 = Enabled.
3	Reserved	0x0	Reserved.
2:1	Port #0 Channel ID	0x0	
0	Port #0 Channel Enable	0x1	0x0 = Disabled. 0x1 = Enabled.



6.2.39.15 CRC8 - 0x0013

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: Sideband Configuration Structure -> Section Length End Section -> Word: Sideband Configuration Structure -> NC-SI Channel to Port Mapping

6.2.40 Thermal Sensor Setting Section Summary Table

Word Offset	Description	Reference
0x0000	Section Header	See section 6.2.40.1
0x0001	PLDM Parameters Pointer	See section 6.2.40.2
0x0002	Thermal Sensor Scripts Pointer	See section 6.2.40.3
0x0003	CRC8	See section 6.2.40.4

6.2.40.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Block Length		Length in: 2 bytes unit - 1 First Section -> Word: Thermal Sensor Setting -> Section Header Last Section -> Word: Thermal Sensor Setting -> CRC8

6.2.40.2 PLDM Parameters Pointer - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Pointer	0x0	Pointer to PLDM.

The Pointer points to the PLDM Parameters section. For PLDM Parameters inner structure detail, see [Section 6.2.41](#).

6.2.40.3 Thermal Sensor Scripts Pointer - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Pointer	0x0	

The Pointer points to the Thermal Sensor Scripts section. For Thermal Sensor Scripts inner structure detail, see [Section 6.2.42](#).



6.2.40.4 CRC8 - 0x0003

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: Thermal Sensor Setting -> Section Header End Section -> Word: Thermal Sensor Setting -> Thermal Sensor Scripts Pointer

6.2.41 PLDM Parameters Section Summary Table

Word Offset	Description	Reference
0x0000	Section Header	See section 6.2.41.1
0x0001	Thermal Flags	See section 6.2.41.2
0x0002	Thermal Resolution - LSB	See section 6.2.41.3
0x0003	Thermal Resolution - MSB	See section 6.2.41.4
0x0004	Thermal Accuracy	See section 6.2.41.5
0x0005	Thermal Sensor Tolerance and Hysteresis	See section 6.2.41.6
0x0006	Thermal Sensor Thresholds	See section 6.2.41.7
0x0007	State Transition Interval	See section 6.2.41.8
0x0008	Update Interval	See section 6.2.41.9
0x0009	Read Limits	See section 6.2.41.10
0x000A	Range Field and Nominal	See section 6.2.41.11
0x000B	Normal Value	See section 6.2.41.12
0x000C	Warning Value	See section 6.2.41.13
0x000D	Critical Value	See section 6.2.41.14
0x000E	Fatal Value	See section 6.2.41.15
0x000F	CRC8	See section 6.2.41.16

6.2.41.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Block Length		Length in: 2 bytes unit - 1 First Section -> Word: PLDM Parameters -> Section Header Last Section -> Word: PLDM Parameters -> CRC8



6.2.41.2 Thermal Flags - 0x0001

Bits	Field Name	Default NVM Value	Description
15:12	TS Interface	0x2	0x0 = I ² C of port 0. 0x1 = I ² C of port 1. 0x2 = I ² C of port 2. 0x3 = I ² C of port 3. 0xF = MNG SMBus.
11:8	Reserved	0x0	Reserved.
7:1	Thermal Sensor SMBus address	0xC	The 7 bit address of the thermal sensor used for PLDM on the TS interface.
0	Thermal Sensor enable	0x1	If set, a thermal sensor PDR and sensor are exposed. 0x0 = Disabled. 0x1 = Enabled.

6.2.41.3 Thermal Resolution - LSB - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Thermal Resolution - LSB	0x0	LSB of resolution in real32.

6.2.41.4 Thermal Resolution - MSB - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	Thermal Resolution - MSB	0x3F80	MSB of resolution in real32.

6.2.41.5 Thermal Accuracy - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	Thermal Accuracy	0x0	Reported thermal sensor accuracy.

6.2.41.6 Thermal Sensor Tolerance and Hysteresis - 0x0005

Bits	Field Name	Default NVM Value	Description
15:8	Tolerance	0x1	Reported thermal sensor plusTolerance and minusTolerance.
7:0	Hysteresis	0x2	Reported thermal sensor hysteresis.

6.2.41.7 Thermal Sensor Thresholds - 0x0006

Bits	Field Name	Default NVM Value	Description
15:14	Reserved	0x0	Reserved.
13	lowerFatal Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
12	lowerWarning Supported	0x0	0x0 = Disabled. 0x1 = Enabled.



Bits	Field Name	Default NVM Value	Description
11	lowerCritical Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
10	upperFatal Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
9	upperCritical Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
8	upperWarning Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
7:0	Threshold And Hysteresis Volatility	0x2	Reported thermal sensor threshold and hysteresis volatility.



6.2.41.8 State Transition Interval - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	State Transition Interval	0x0	Reported thermal sensor state transition interval in ms; should be translated to real32 and to seconds.

6.2.41.9 Update Interval - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	Update Interval	0x0	Reported thermal sensor update interval in seconds; should be translated to real32.

6.2.41.10 Read Limits - 0x0009

Bits	Field Name	Default NVM Value	Description
15:8	Max Readable	0xFF	Reported thermal sensor maxReadable.
7:0	Min Readable	0x0	Reported thermal sensor minReadable.

6.2.41.11 Range Field and Nominal - 0x000A

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	Reserved
14	fatalLow Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
13	fatalHigh Supported	0x1	0x0 = Disabled. 0x1 = Enabled.
12	criticalLow Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
11	criticalHigh Supported	0x1	0x0 = Disabled. 0x1 = Enabled.
10	normalLow Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
9	normalHigh Supported	0x1	0x0 = Disabled. 0x1 = Enabled.
8	nominalValue Supported	0x0	0x0 = Disabled. 0x1 = Enabled.
7:0	nominalValue	0x0	Reported thermal sensor nominalValue.

6.2.41.12 Normal Value - 0x000B

Bits	Field Name	Default NVM Value	Description
15:8	normalMax	0x5A	Reported thermal sensor normalMax.
7:0	normalMin	0x0	Reported thermal sensor normalMin.



6.2.41.13 Warning Value - 0x000C

Bits	Field Name	Default NVM Value	Description
15:8	warningHigh	0x5F	Reported thermal sensor warningHigh.
7:0	warningLow	0x0	Reported thermal sensor warningLow.

6.2.41.14 Critical Value - 0x000D

Bits	Field Name	Default NVM Value	Description
15:8	criticalHigh	0x69	Reported thermal sensor criticalHigh.
7:0	criticalLow	0x0	Reported thermal sensor criticalLow.

6.2.41.15 Fatal Value - 0x000E

Bits	Field Name	Default NVM Value	Description
15:8	fatalHigh	0x0	Reported thermal sensor fatalHigh.
7:0	fatalLow	0x0	Reported thermal sensor fatalLow.

6.2.41.16 CRC8 - 0x000F

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: PLDM Parameters -> Section Header End Section -> Word: PLDM Parameters -> Fatal Value

6.2.42 Thermal Sensor Scripts Section Summary Table

Word Offset	Description	Reference
0x0000	Section Header	See section 6.2.42.1
0x0001	TS_CNT + I2C PRT	See section 6.2.42.2
0x0002	ALERT_SCRIPT_SIZE + ALERT_SCRIPT_OFF	See section 6.2.42.3
0x0003	PLDM_SCRIPT_SIZE + PLDM_SCRIPT_OFF	See section 6.2.42.4
0x0004	READ_SCRIPT_SIZE + READ_SCRIPT_OFF	See section 6.2.42.5
0x0005	Register 0 Address	See section 6.2.42.6
0x0006	Register 0 Data	See section 6.2.42.7
0x0007	Register 1 Address	See section 6.2.42.8
0x0008	Register 1 Data	See section 6.2.42.9
0x0009	Register 2 Address	See section 6.2.42.10
0x000A	Register 2 Data	See section 6.2.42.11



Word Offset	Description	Reference
0x000B	Register 3 Address	See section 6.2.42.12
0x000C	Register 3 Data	See section 6.2.42.13
0x000D	Register 4 Address	See section 6.2.42.14
0x000E	Register 4 Data	See section 6.2.42.15
0x000F	Register 5 Address	See section 6.2.42.16
0x0010	Register 5 Data	See section 6.2.42.17
0x0011	Register 6 Address	See section 6.2.42.18
0x0012	Register 6 Data	See section 6.2.42.19
0x0013	Register 7 Address	See section 6.2.42.20
0x0014	Register 7 Data	See section 6.2.42.21
0x0015	Register 8 Address	See section 6.2.42.22
0x0016	Register 8 Data	See section 6.2.42.23
0x0017	Register 0 Address	See section 6.2.42.24
0x0018	Register 0 Data	See section 6.2.42.25
0x0019	Register 1 Address	See section 6.2.42.26
0x001A	Register 1 Data	See section 6.2.42.27
0x001B	Register 2 Address	See section 6.2.42.28
0x001C	Register 2 Data	See section 6.2.42.29
0x001D	Register 3 Address	See section 6.2.42.30
0x001E	Register 3 Data	See section 6.2.42.31
0x001F	Register 4 Address	See section 6.2.42.32
0x0020	Register 4 Data	See section 6.2.42.33
0x0021	Register 5 Address	See section 6.2.42.34
0x0022	Register 5 Data	See section 6.2.42.35
0x0023	Register 6 Address	See section 6.2.42.36
0x0024	Register 6 Data	See section 6.2.42.37
0x0025	Register 7 Address	See section 6.2.42.38
0x0026	Register 7 Data	See section 6.2.42.39
0x0027	Register 8 Address	See section 6.2.42.40
0x0028	Register 8 Data	See section 6.2.42.41
0x0029	Register 0 Address	See section 6.2.42.42
0x002A	Register 1 Address	See section 6.2.42.43
0x002B	Register 2 Address	See section 6.2.42.44
0x002C	CRC8	See section 6.2.42.45



6.2.42.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Block Length		Length in: 2 bytes unit - 1 First Section -> Word: Thermal Sensor Scripts -> Section Header Last Section -> Word: Thermal Sensor Scripts -> CRC8

6.2.42.2 TS_CNT + I2C_PRT - 0x0001

Bits	Field Name	Default NVM Value	Description
15:8	I2C_PRT	0x2	The I ² C port used to communicate with the thermal sensors. Values of 0x0 through 0x3 are reserved for I ² C port 0 through 3. Value of 0xF is reserved for the MNG SMBus port.
7:0	TS_CNT	0x3	Number of implemented thermal sensors (0x0 for no thermal sensors on the board).

6.2.42.3 ALERT_SCRIPT_SIZE + ALERT_SCRIPT_OFF - 0x0002

Bits	Field Name	Default NVM Value	Description
15:8	ALERT_SCRIPT_OFF	0x5	Word offset of the alert script (relative to the beginning of this table).
7:0	ALERT_SCRIPT_SIZE	0x9	Number of registers to be accessed (address / data couple) when alert mode is selected.

6.2.42.4 PLDM_SCRIPT_SIZE + PLDM_SCRIPT_OFF - 0x0003

Bits	Field Name	Default NVM Value	Description
15:8	PLDM_SCRIPT_OFF	0x17	Word offset of the PLDM script (relative to the beginning of this table).
7:0	PLDM_SCRIPT_SIZE	0x9	Number of registers to be accessed when PLDM mode is selected.

6.2.42.5 READ_SCRIPT_SIZE + READ_SCRIPT_OFF - 0x0004

Bits	Field Name	Default NVM Value	Description
15:8	READ_SCRIPT_OFF	0x29	Word offset of the read script (relative to the beginning of this table).
7:0	READ_SCRIPT_SIZE	0x3	Number of registers to be accessed for read temperature from the thermal sensors (used in PLDM and debug AQC).

6.2.42.6 Register 0 Address - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Register 0 Address	0x9001	



6.2.42.7 Register 0 Data - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	Register 0 Data	0x0	Sensor enabled with an active low alert signal.

6.2.42.8 Register 1 Address - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	Register 1 Address	0x9003	High threshold register at offset 0x3 for the T-control sensor at I ² C address 0x48.

6.2.42.9 Register 1 Data - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	Register 1 Data	0x5A00	T-Control is set to 90 °C.

6.2.42.10 Register 2 Address - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	Register 2 Address	0x9002	Low threshold register at offset 0x2 for the T-control sensor at I ² C address 0x48.

6.2.42.11 Register 2 Data - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	Register 2 Data	0x5500	T-control minus 5 °C for hysteresis.

6.2.42.12 Register 3 Address - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	Register 3 Address	0x9201	Control register at offset 0x1 of the T-warning sensor at I ² C address 0x49.

6.2.42.13 Register 3 Data - 0x000C

Bits	Field Name	Default NVM Value	Description
15:0	Register 3 Data	0x0	Sensor enabled with a low active alert signal.

6.2.42.14 Register 4 Address - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	Register 4 Address	0x9203	High threshold register at offset 0x3 for the T-warning sensor at I ² C address 0x49.



6.2.42.15 Register 4 Data - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	Register 4 Data	0x5F00	T-warning is set to 95 °C.

6.2.42.16 Register 5 Address - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	Register 5 Address	0x9202	Low threshold register at offset 0x2 for the T-warning sensor at I ² C address 0x49.

6.2.42.17 Register 5 Data - 0x0010

Bits	Field Name	Default NVM Value	Description
15:0	Register 5 Data	0x5A00	T-warning minus 5 °C for hysteresis.

6.2.42.18 Register 6 Address - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	Register 6 Address	0x9401	Control register at offset 0x1 of the T-critical sensor at I ² C address 0x4A.

6.2.42.19 Register 6 Data - 0x0012

Bits	Field Name	Default NVM Value	Description
15:0	Register 6 Data	0x0	Sensor enabled with a low active alert signal.

6.2.42.20 Register 7 Address - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	Register 7 Address	0x9403	High threshold register at offset 0x3 for the T-critical sensor at I ² C address 0x4A.

6.2.42.21 Register 7 Data - 0x0014

Bits	Field Name	Default NVM Value	Description
15:0	Register 7 Data	0x6900	T-critical is set to 105 °C.

6.2.42.22 Register 8 Address - 0x0015

Bits	Field Name	Default NVM Value	Description
15:0	Register 8 Address	0x9402	Low threshold register at offset 0x2 for the T-critical sensor at I ² C address 0x4A.

**6.2.42.23 Register 8 Data - 0x0016**

Bits	Field Name	Default NVM Value	Description
15:0	Register 8 Data	0x6400	T-critical minus 5 °C for hysteresis.

6.2.42.24 Register 0 Address - 0x0017

Bits	Field Name	Default NVM Value	Description
15:0	Register 0 Address	0x9001	

6.2.42.25 Register 0 Data - 0x0018

Bits	Field Name	Default NVM Value	Description
15:0	Register 0 Data	0x0	Sensor enabled with an active low alert signal.

6.2.42.26 Register 1 Address - 0x0019

Bits	Field Name	Default NVM Value	Description
15:0	Register 1 Address	0x9003	High threshold register at offset 0x3 for the T-control sensor at I ² C address 0x48.

6.2.42.27 Register 1 Data - 0x001A

Bits	Field Name	Default NVM Value	Description
15:0	Register 1 Data	0x5A00	T-control is set to 90 °C.

6.2.42.28 Register 2 Address - 0x001B

Bits	Field Name	Default NVM Value	Description
15:0	Register 2 Address	0x9002	Low threshold register at offset 0x2 for the T-control sensor at I ² C address 0x48.

6.2.42.29 Register 2 Data - 0x001C

Bits	Field Name	Default NVM Value	Description
15:0	Register 2 Data	0x5500	T-control minus 5 °C for hysteresis.

6.2.42.30 Register 3 Address - 0x001D

Bits	Field Name	Default NVM Value	Description
15:0	Register 3 Address	0x9201	Control register at offset 0x1 of the T-warning sensor at I ² C address 0x49.



6.2.42.31 Register 3 Data - 0x001E

Bits	Field Name	Default NVM Value	Description
15:0	Register 3 Data	0x0	Sensor enabled with a low active alert signal.

6.2.42.32 Register 4 Address - 0x001F

Bits	Field Name	Default NVM Value	Description
15:0	Register 4 Address	0x9203	High threshold register at offset 0x3 for the T-warning sensor at I ² C address 0x49.

6.2.42.33 Register 4 Data - 0x0020

Bits	Field Name	Default NVM Value	Description
15:0	Register 4 Data	0x5F00	T-warning is set to 95 °C.

6.2.42.34 Register 5 Address - 0x0021

Bits	Field Name	Default NVM Value	Description
15:0	Register 5 Address	0x9202	Low threshold register at offset 0x2 for the T-warning sensor at I ² C address 0x49.

6.2.42.35 Register 5 Data - 0x0022

Bits	Field Name	Default NVM Value	Description
15:0	Register 5 Data	0x5A00	T-warning minus 5 °C for hysteresis.

6.2.42.36 Register 6 Address - 0x0023

Bits	Field Name	Default NVM Value	Description
15:0	Register 6 Address	0x9401	Control register at offset 0x1 of the T-critical sensor at I ² C address 0x4A.

6.2.42.37 Register 6 Data - 0x0024

Bits	Field Name	Default NVM Value	Description
15:0	Register 6 Data	0x0	Sensor enabled with a low active alert signal.

6.2.42.38 Register 7 Address - 0x0025

Bits	Field Name	Default NVM Value	Description
15:0	Register 7 Address	0x9403	High threshold register at offset 0x3 for the T-critical sensor at I ² C address 0x4A.



6.2.42.39 Register 7 Data - 0x0026

Bits	Field Name	Default NVM Value	Description
15:0	Register 7 Data	0x6900	T-critical is set to 105 °C.

6.2.42.40 Register 8 Address - 0x0027

Bits	Field Name	Default NVM Value	Description
15:0	Register 8 Address	0x9402	Low threshold register at offset 0x2 for the T-critical sensor at I ² C address 0x4A.

6.2.42.41 Register 8 Data - 0x0028

Bits	Field Name	Default NVM Value	Description
15:0	Register 8 Data	0x6400	T-critical minus 5 °C for hysteresis.

6.2.42.42 Register 0 Address - 0x0029

Bits	Field Name	Default NVM Value	Description
15:0	Register 0 Address	0x9000	Temperature register at offset 0x0 of the T-control sensor at I ² C address 0x48.

6.2.42.43 Register 1 Address - 0x002A

Bits	Field Name	Default NVM Value	Description
15:0	Register 1 Address	0x9200	Temperature register at offset 0x0 of the T-warning sensor at I ² C address 0x49.

6.2.42.44 Register 2 Address - 0x002B

Bits	Field Name	Default NVM Value	Description
15:0	Register 2 Address	0x9400	Temperature register at offset 0x0 of the T-critical sensor at I ² C address 0x4A.

6.2.42.45 CRC8 - 0x002C

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: Thermal Sensor Scripts -> Section Header End Section -> Word: Thermal Sensor Scripts -> Register 2 Address



6.2.43 EMP Settings Module Header Section Summary Table

This section contains the modes of operation of the EMP.

Word Offset	Description	Reference
0x15000	Section Length	See section 6.2.43.1
0x15001	Common Firmware Parameters	See section 6.2.43.2
0x15002	FW Misc	See section 6.2.43.3
0x15003	Features Enable - Copy	See section 6.2.43.4
0x15005	EEE Variables	See section 6.2.43.5
0x15006	LLDP Configuration Pointer	See section 6.2.43.6
0x1500B	LLDP TLVs Pointer	See section 6.2.43.11
0x1500C	Reserved	See section 6.2.43.7
0x1500D	PXE PFC Timer Value	See section 6.2.43.8
0x1500E	PXE GPC High Threshold Value	See section 6.2.43.9
0x1500F	PXE GPC Low Threshold Value	See section 6.2.43.10
0x15010	CRC8	See section 6.2.43.11

6.2.43.1 Section Length - 0x15000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.



6.2.43.2 Common Firmware Parameters - 0x15001

Bits	Field Name	Default NVM Value	Description
15:10	PFR Timeout Value	0x12	This field defines the amount of time (in 10-ms units) that firmware waits for hardware to complete a PF reset before escalating to a core reset. If this field is 0x0, the timeout is disabled.
9	PF Reset On Queue Overflow	0x0	0x0 = Disabled. 0x1 Enabled.
8:0	Reserved	0x3	Reserved.

6.2.43.3 FW Misc - 0x15002

Bits	Field Name	Default NVM Value	Description
15:0	FW Reserved	0x0001	Reserved.

6.2.43.4 Features Enable - Copy - 0x15003

This is a copy of features enable in EMP SR Settings Module Header - Features enable. Note that this is the legacy place of the word, the word moved to the new place and a copy is kept for backward compatibility.

Bits	Field Name	Default NVM Value	Description
15:13	Reserved	0x0	Reserved.
12	Channel Identifier	0x0	Defines the Ethertype of the channel identifier used to differentiate channels within a port extender. 0x0 = S-tag. 0x1 = VLAN.
11:9	Switching Mode	0x0	0x0 = EVB switching 802.1Qbg switching.
8	Proxy Enabled for Port 3	0x1	If cleared, the proxy functionality embedded in the EMP is disabled on Port 3. It could be re-enabled on this port only by resetting this bit to 1b and performing an EMPR cycle. 0x0 = Disabled. 0x1 Enabled.
7	Proxy Enabled for Port 2	0x1	If cleared, the proxy functionality embedded in the EMP is disabled on Port 2. It could be re-enabled on this port only by resetting this bit to 1b and performing an EMPR cycle. 0x0 = Disabled. 0x1 Enabled.
6	Proxy Enabled for Port 1	0x1	If cleared, the proxy functionality embedded in the EMP is disabled on Port 1. It could be re-enabled on this port only by resetting this bit to 1b and performing an EMPR cycle. 0x0 = Disabled. 0x1 Enabled.
5	Proxy Enabled for Port 0	0x1	If cleared, the proxy functionality embedded in the EMP is disabled on Port 0. It could be re-enabled on this port only by resetting this bit to 1b and performing an EMPR cycle. 0x0 = Disabled. 0x1 Enabled.
4:2	Reserved	0x0	Reserved.
1	EVB Protocols Enabled	0x1	If set, filtering according to IEEE 802.1QBg specification is enabled. 0x0 = Disabled. 0x1 =Enabled.



Bits	Field Name	Default NVM Value	Description
0	VEB Statistics Disable	0x1	When clear (default), VEB statistics are enabled. When set, VEB statistics are disabled.

6.2.43.5 EEE Variables - 0x15005

Bits	Field Name	Default NVM Value	Description
15:8	receiveTW	0xE	Value determines the time (expressed in microseconds) that the receiving link partner is requesting the transmitting link partner to wait before starting the transmission data following the LPI. This value is platform dependent and depends on the OEM platform.
7:0	transmitTW	0xE	Value determines the time (expressed in microseconds) that the transmitting link partner waits before it starts transmitting data after leaving the LPI mode. This value is platform dependent and depends on the OEM platform.

6.2.43.6 LLDP Configuration Pointer - 0x15006

Bits	Field Name	Default NVM Value	Description
15:0	LLDP Configuration Pointer	0xFFFF	

The LLDP configuration pointer points to the LLDP Configuration section. For LLDP Configuration inner structure detail, see [Section 6.2.44](#).



6.2.43.7 Reserved - 0x1500C

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.43.8 PXE PFC Timer Value - 0x1500D

Bits	Field Name	Default NVM Value	Description
15:14	Reserved	0x0	Reserved.
13:0	PXE PFC Timer Value	0x4	

6.2.43.9 PXE GPC High Threshold Value - 0x1500E

Bits	Field Name	Default NVM Value	Description
15:0	PXE GPC High Threshold	0x80	

6.2.43.10 PXE GPC Low Threshold Value - 0x1500F

Bits	Field Name	Default NVM Value	Description
15:0	PXE GPC Low Threshold	0x10	

6.2.43.11 CRC8 - 0x15010

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: EMP Settings Module Header -> Section Length End Section -> Word: EMP Settings Module Header -> PXE GPC Low Threshold

6.2.44 LLDP Configuration Section Summary Table

Default settings to the embedded LLDP agent.

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.44.1
0x0001	LLDP Admin Status	See section 6.2.44.2
0x0002	msgFastTx	See section 6.2.44.3
0x0003	msgTxInterval	See section 6.2.44.4
0x0004	LLDP Tx Parameters	See section 6.2.44.5
0x0005	LLDP Initialization Timers	See section 6.2.44.6



Word Offset	Description	Reference
0x0006	ENDLESS_XOFF_THRESH	See section 6.2.44.7
0x0007	DCBX Mode	See section 6.2.44.8
0x0008	CRC8	See section 6.2.44.9

6.2.44.1 Section Length - 0x0000

The length of the section in words. Note that section length does not include a count for the section length word.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.



6.2.44.2 LLDP Admin Status - 0x0001

Defines status of an LLDP agent. Each LAN port has independent status.

- 3: Both receive and transmit enabled.
- 2: LLDP is configured for transmits only.
- 1: LLDP is configured for receives only.
- 0: LLDP agent is disabled.

Bits	Field Name	Default NVM Value	Description
15:12	Port 3	0x3	Defines the status of an LLDP agent. Applies to LAN Port 3.
11:8	Port 2	0x3	Defines the status of an LLDP agent. Applies to LAN Port 2.
7:4	Port 1	0x3	Defines the status of an LLDP agent. Applies to LAN Port 1.
3:0	Port 0	0x3	Defines the status of an LLDP agent. Applies to LAN Port 0.

6.2.44.3 msgFastTx - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	msgFastTx	0x1	Time interval in timer ticks (seconds) between PDU transmits during fast transmits period.

6.2.44.4 msgTxInterval - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	msgTxInterval	0x1E	Time in timer ticks (seconds) between transmissions during normal transmission.

6.2.44.5 LLDP Tx Parameters - 0x0004

Bits	Field Name	Default NVM Value	Description
15:8	txCreditMax	0x5	Determines the maximum number of LLDPDUs that can be sent per second.
7:0	msgTxHold	0x4	Used as a multiplier of <i>msgTxInterval</i> to determine the <i>txTTL</i> that is carried in the LLDP frames.

6.2.44.6 LLDP Initialization Timers - 0x0005

Timers used by an LLDP agent during initialization and when to reinitialize. All times are measured in seconds.

Bits	Field Name	Default NVM Value	Description
15:8	reinitDelay	0x2	This parameter indicates the amount of delay, in seconds, from when <i>AdminStatus</i> becomes disabled until re-initialization is attempted.
7:0	txFastInit	0x4	This variable is used as the initial value for the <i>txFast</i> variable. This value determines the number of LLDPDUs that are transmitted during a fast transmission period.



6.2.44.7 ENDLESS_XOFF_THRESH - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	ENDLESS_XOFF_THRESH	0xA	Define a time limit that firmware waits for an XOFF condition during PFR flow. The time is defined in 1 ms units. A possible default setting is 10 ms.

6.2.44.8 DCBX Mode - 0x0007

Defines per-PORT DCBX mode. Note that each port can be configured independently regardless of the other ports configurations.

- 0: No DCBX is supported.
- 1: IEEE DCBX only.
- 3: Auto-Select, starting in IEEE (default).
- 2: CEE DCBX only.

Bits	Field Name	Default NVM Value	Description
15:12	DCBX Port[3]	0x3	Port[0] DCBX Mode. 0x0 = No DCBX support. 0x1 = IEEE DCBX only. 0x2 = CEE DCBX only. 0x3 = Auto-select starting in IEEE (default). Note: This field is preserved by the Intel NVM update tool.
11:8	DCBX Port[2]	0x3	Port[0] DCBX Mode. 0x0 = No DCBX support. 0x1 = IEEE DCBX only. 0x2 = CEE DCBX only. 0x3 = Auto-select starting in IEEE (default). Note: This field is preserved by the Intel NVM update tool.
7:4	DCBX Port[1]	0x3	Port[0] DCBX Mode. 0x0 = No DCBX support. 0x1 = IEEE DCBX only. 0x2 = CEE DCBX only. 0x3 = Auto-select starting in IEEE (default). Note: This field is preserved by the Intel NVM update tool.
3:0	DCBX Port[0]	0x3	Port[0] DCBX Mode. 0x0 = No DCBX support. 0x1 = IEEE DCBX only. 0x2 = CEE DCBX only. 0x3 = Auto-select starting in IEEE (default). Note: This field is preserved by the Intel NVM update tool.

6.2.44.9 CRC8 - 0x0008

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.



Bits	Field Name	Default NVM Value	Description
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: LLDP Configuration -> Section Length End Section -> Word: LLDP Configuration -> DCBX Mode



6.2.45 2nd Free Provisioning Area Section Summary Table

Free area used as the new bank when updating 8 KB long modules which are mapped outside the shadow RAM.

Word Offset	Description	Reference
0x16000	[New Word]	See section 6.2.45.1

6.2.45.1 [New Word] - 0x16000

Raw data module length: variable

6.2.46 OROM Section Summary Table

Option ROM module.

It contains pre-boot code and settings read by BIOS.

Word Offset	Description	Reference
0x0000	OROM Data	See section 6.2.46.1
0x0001	moduleTypeL	See section 6.2.46.2
0x0002	moduleTypeH	See section 6.2.46.3
0x0003	headerLenL	See section 6.2.46.4
0x0004	headerLenH	See section 6.2.46.5
0x0005	headerVersionL	See section 6.2.46.6
0x0006	headerVersionH	See section 6.2.46.7
0x0007	moduleIDL	See section 6.2.46.8
0x0008	moduleIDH	See section 6.2.46.9
0x0009	moduleVendorL	See section 6.2.46.10
0x000A	moduleVendorH	See section 6.2.46.11
0x000B	dateL	See section 6.2.46.12
0x000C	dateH	See section 6.2.46.13
0x000D	sizeL	See section 6.2.46.14
0x000E	sizeH	See section 6.2.46.15
0x000F	keySizeL	See section 6.2.46.16
0x0010	keySizeH	See section 6.2.46.17
0x0011	modulusSizeL	See section 6.2.46.18
0x0012	modulusSizeH	See section 6.2.46.19
0x0013	exponentSizeL	See section 6.2.46.20
0x0014	exponentSizeH	See section 6.2.46.21
0x0015	lad_srevL	See section 6.2.46.22
0x0016	lad_srevH	See section 6.2.46.23
0x0017	reserved1L	See section 6.2.46.24



Word Offset	Description	Reference
0x0018	reserved1H	See section 6.2.46.25
0x0019	lad_fw_entry_offsetL	See section 6.2.46.26
0x001A	lad_fw_entry_offsetH	See section 6.2.46.27
0x001B	reserved2L	See section 6.2.46.28
0x001C	reserved2H	See section 6.2.46.29
0x001D	lad_image_unique_idL	See section 6.2.46.30
0x001E	lad_image_unique_idH	See section 6.2.46.31
0x001F	lad_module_idL	See section 6.2.46.32
0x0020	lad_module_idH	See section 6.2.46.33
0x0021 + 1*n, n=0...31	Reserved	See section 6.2.46.34
0x0041 + 1*n, n=0...127	RSA Public Key	See section 6.2.46.35
0x00C1	RSA ExponentL	See section 6.2.46.36
0x00C2	RSA ExponentH	See section 6.2.46.37
0x00C3 + 1*n, n=0...127	Encrypted SHA256 Hash	See section 6.2.46.38
0x0143	Device Blank NVM Device ID	See section 6.2.46.39
0x0144	Max Module AreaL	See section 6.2.46.40
0x0145	Max Module AreaH	See section 6.2.46.41
0x0146	Current Module AreaL	See section 6.2.46.42
0x0147	Current Module AreaH	See section 6.2.46.43
0x0148	Format Version + CRC	See section 6.2.46.44
0x0149	Code Revision	See section 6.2.46.45
0x014A	Reserved Spare Word	See section 6.2.46.46

6.2.46.1 OROM Data - 0x0000

Raw data module length: variable



6.2.46.2 moduleTypeL - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	moduleType	0x6	

6.2.46.3 moduleTypeH - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	moduleTypeH		

6.2.46.4 headerLenL - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	headerLenL	0xA1	

6.2.46.5 headerLenH - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	headerLenH		

6.2.46.6 headerVersionL - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionL	0x00010000	

6.2.46.7 headerVersionH - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionH		

6.2.46.8 moduleIDL - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	moduleIDL	0x0	

6.2.46.9 moduleIDH - 0x0008

Bits	Field Name	Default NVM Value	Description
15	signMode	0x0	
14:0	moduleIDH		



6.2.46.10 moduleVendorL - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorL	0x00008086	

6.2.46.11 moduleVendorH - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorH		

6.2.46.12 dateL - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	DateL	0x20130530	0xMMDD.

6.2.46.13 dateH - 0x000C

Bits	Field Name	Default NVM Value	Description
15:0	DateH		0xYYYY.



6.2.46.14 sizeL - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	sizeL	0x00048800	

6.2.46.15 sizeH - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	sizeH		

6.2.46.16 keySizeL - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	keySizeL	0x40	

6.2.46.17 keySizeH - 0x0010

Bits	Field Name	Default NVM Value	Description
15:0	keySizeH		

6.2.46.18 modulusSizeL - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeL	0x40	

6.2.46.19 modulusSizeH - 0x0012

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeH		

6.2.46.20 exponentSizeL - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeL	0x1	

6.2.46.21 exponentSizeH - 0x0014

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeH		



6.2.46.22 lad_srevL - 0x0015

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevL	0x0	

6.2.46.23 lad_srevH - 0x0016

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevH		

6.2.46.24 reserved1L - 0x0017

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.2.46.25 reserved1H - 0x0018

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.2.46.26 lad_fw_entry_offsetL - 0x0019

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetL	0x14C	

6.2.46.27 lad_fw_entry_offsetH - 0x001A

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetH		

6.2.46.28 reserved2L - 0x001B

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.2.46.29 reserved2H - 0x001C

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	



6.2.46.30 lad_image_unique_idL - 0x001D

Bits	Field Name	Default NVM Value	Description
15:0	lad_image_unique_idL	0x0	

6.2.46.31 lad_image_unique_idH - 0x001E

Bits	Field Name	Default NVM Value	Description
15:0	lad_image_unique_idH		

6.2.46.32 lad_module_idL - 0x001F

Bits	Field Name	Default NVM Value	Description
15:0	lad_module_idL	0x5	0x1 = EMP image. 0x2 = PE image. 0x3 = PCIe analog. 0x4 = PHY analog. 0x5 = Option ROM.

6.2.46.33 lad_module_idH - 0x0020

Bits	Field Name	Default NVM Value	Description
15:0	lad_module_idH		

6.2.46.34 reserved[n] - 0x0021 + 1*n, n=0...31

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.2.46.35 RSA Public Key [n] - 0x0041 + 1*n, n=0...127

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.2.46.36 RSA ExponentL - 0x00C1

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentL	0x0	



6.2.46.37 RSA ExponentH - 0x00C2

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentH	0x0	

6.2.46.38 Encrypted SHA256 Hash[n] - 0x00C3 + 1*n, n=0...127

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.2.46.39 Device Blank NVM Device ID - 0x0143

Bits	Field Name	Default NVM Value	Description
15:0	Device Blank NVM Device ID	0x154B	

6.2.46.40 Max Module AreaL - 0x0144

Bits	Field Name	Default NVM Value	Description
15:0	Max Module AreaL	0x1000	

6.2.46.41 Max Module AreaH - 0x0145

Bits	Field Name	Default NVM Value	Description
15:0	Max Module AreaH	0x0009	

6.2.46.42 Current Module AreaL - 0x0146

Bits	Field Name	Default NVM Value	Description
15:0	Current Module AreaL	0x1000	

6.2.46.43 Current Module AreaH - 0x0147

Bits	Field Name	Default NVM Value	Description
15:0	Current Module AreaH	0x0000	

6.2.46.44 Format Version + CRC - 0x0148

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x0	0x0 = CRC not used. 0x1 = CRC used.



Bits	Field Name	Default NVM Value	Description
14:8	Format Version	0x2	
7:0	CRC8	0xFF	

6.2.46.45 Code Revision - 0x0149

Bits	Field Name	Default NVM Value	Description
15:8	Major Revision	0x0	
7:0	Minor Revision	0x0	

6.2.46.46 Reserved Spare Word - 0x014A

Bits	Field Name	Default NVM Value	Description
15:0	Reserved Spare Word	0x0	

Bits	Field Name	Default NVM Value	Description
15:0	DateL	0x20130530	0xMMDD.

Bits	Field Name	Default NVM Value	Description
15:0	DateH		0xYYYY.

Raw data module length: variable



6.2.47 EMP Image Section Summary Table

Word Offset	Description	Reference
0x0000	moduleTypeL	See section 6.2.47.1
0x0001	moduleTypeH	See section 6.2.47.2
0x0002	headerLenL	See section 6.2.47.3
0x0003	headerLenH	See section 6.2.47.4
0x0004	headerVersionL	See section 6.2.47.5
0x0005	headerVersionH	See section 6.2.47.6
0x0006	moduleIDL	See section 6.2.47.7
0x0007	moduleIDH	See section 6.2.47.8
0x0008	moduleVendorL	See section 6.2.47.9
0x0009	moduleVendorH	See section 6.2.47.10
0x000A	dateL	See section 6.2.47.11
0x000B	dateH	See section 6.2.47.12
0x000C	sizeL	See section 6.2.47.13
0x000D	sizeH	See section 6.2.47.14
0x000E	keySizeL	See section 6.2.47.15
0x000F	keySizeH	See section 6.2.47.16
0x0010	modulusSizeL	See section 6.2.47.17
0x0011	modulusSizeH	See section 6.2.47.18
0x0012	exponentSizeL	See section 6.2.47.19
0x0013	exponentSizeH	See section 6.2.47.20
0x0014	lad_srevL	See section 6.2.47.21
0x0015	lad_srevH	See section 6.2.47.22
0x0016	reserved1L	See section 6.2.47.23
0x0017	reserved1H	See section 6.2.47.24
0x0018	lad_fw_entry_offsetL	See section 6.2.47.25
0x0019	lad_fw_entry_offsetH	See section 6.2.47.26
0x001A	reserved2L	See section 6.2.47.27
0x001B	reserved2H	See section 6.2.47.28
0x001C	lad_image_unique_idL	See section 6.2.47.29
0x001D	lad_image_unique_idH	See section 6.2.47.30
0x001E	lad_module_idL	See section 6.2.47.31
0x001F	lad_module_idH	See section 6.2.47.32
0x0020 + 1*n, n=0...31	Reserved	See section 6.2.47.33
0x0040 + 1*n, n=0...127	RSA Public Key	See section 6.2.47.34
0x00C0	RSA ExponentL	See section 6.2.47.35
0x00C1	RSA ExponentH	See section 6.2.47.36



Word Offset	Description	Reference
0x00C2 + 1*n, n=0...127	Encrypted SHA256 Hash	See section 6.2.47.37
0x0142	Device Blank NVM Device ID	See section 6.2.47.38
0x0143	Max Module AreaL	See section 6.2.47.39
0x0144	Max Module AreaH	See section 6.2.47.40
0x0145	Current Module AreaL	See section 6.2.47.41
0x0146	Current Module AreaH	See section 6.2.47.42
0x0147	Format Version + CRC	See section 6.2.47.43
0x0148	Code Revision	See section 6.2.47.44
0x0149	Reserved Spare Word	See section 6.2.47.45
0x014A	EMP Image Raw Data	See section 6.2.47.46
0x014B	RO Trailer	See section 6.2.47.47



6.2.47.1 moduleTypeL - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	moduleType	0x6	

6.2.47.2 moduleTypeH - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	moduleTypeH		

6.2.47.3 headerLenL - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	headerLenL	0xA1	

6.2.47.4 headerLenH - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	headerLenH		

6.2.47.5 headerVersionL - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionL	0x00010000	

6.2.47.6 headerVersionH - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionH		

6.2.47.7 moduleIDL - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	moduleIDL	0x0	

6.2.47.8 moduleIDH - 0x0007

Bits	Field Name	Default NVM Value	Description
15	signMode	0x0	
14:0	moduleIDH		



6.2.47.9 moduleVendorL - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorL	0x00008086	

6.2.47.10 moduleVendorH - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorH		

6.2.47.11 dateL - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	DateL	0x20130530	0xMMDD.

6.2.47.12 dateH - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	DateH		0xYYYY.



6.2.47.13 sizeL - 0x000C

Bits	Field Name	Default NVM Value	Description
15:0	sizeL	0x00048800	

6.2.47.14 sizeH - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	sizeH		

6.2.47.15 keySizeL - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	keySizeL	0x40	

6.2.47.16 keySizeH - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	keySizeH		

6.2.47.17 modulusSizeL - 0x0010

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeL	0x40	

6.2.47.18 modulusSizeH - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeH		

6.2.47.19 exponentSizeL - 0x0012

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeL	0x1	

6.2.47.20 exponentSizeH - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeH		



6.2.47.21 lad_srevL - 0x0014

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevL	0x1	

6.2.47.22 lad_srevH - 0x0015

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevH		

6.2.47.23 reserved1L - 0x0016

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.2.47.24 reserved1H - 0x0017

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.2.47.25 lad_fw_entry_offsetL - 0x0018

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetL	0x14C	

6.2.47.26 lad_fw_entry_offsetH - 0x0019

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetH		

6.2.47.27 reserved2L - 0x001A

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.2.47.28 reserved2H - 0x001B

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	



6.2.47.29 lad_image_unique_idL - 0x001C

Bits	Field Name	Default NVM Value	Description
15:0	lad_image_unique_idL	0x0	

6.2.47.30 lad_image_unique_idH - 0x001D

Bits	Field Name	Default NVM Value	Description
15:0	lad_image_unique_idH		

6.2.47.31 lad_module_idL - 0x001E

Bits	Field Name	Default NVM Value	Description
15:0	lad_module_idL	0x1	0x1 = EMP image. 0x2 = PE image. 0x3 = PCIe analog. 0x4 = PHY analog. 0x5 = Option ROM.

6.2.47.32 lad_module_idH - 0x001F

Bits	Field Name	Default NVM Value	Description
15:0	lad_module_idH		

6.2.47.33 reserved[n] - 0x0020 + 1*n, n=0...31

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.2.47.34 RSA Public Key [n] - 0x0040 + 1*n, n=0...127

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.2.47.35 RSA ExponentL - 0x00C0

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentL	0x0	



6.2.47.36 RSA ExponentH - 0x00C1

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentH	0x0	

6.2.47.37 Encrypted SHA256 Hash[n] - 0x00C2 + 1*n, n=0...127

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.2.47.38 Device Blank NVM Device ID - 0x0142

Bits	Field Name	Default NVM Value	Description
15:0	Device Blank NVM Device ID	0x154B	

6.2.47.39 Max Module AreaL - 0x0143

Bits	Field Name	Default NVM Value	Description
15:0	Max Module AreaL	0x1000	

6.2.47.40 Max Module AreaH - 0x0144

Bits	Field Name	Default NVM Value	Description
15:0	Max Module AreaH	0x0009	

6.2.47.41 Current Module AreaL - 0x0145

Bits	Field Name	Default NVM Value	Description
15:0	Current Module AreaL	0xE000	

6.2.47.42 Current Module AreaH - 0x0146

Bits	Field Name	Default NVM Value	Description
15:0	Current Module AreaH	0x0006	

6.2.47.43 Format Version + CRC - 0x0147

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.



Bits	Field Name	Default NVM Value	Description
14:8	Format Version	0x2	
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: EMP Image -> moduleTypeL End Section -> Word: EMP Image -> RO Trailer

6.2.47.44 Code Revision - 0x0148

Bits	Field Name	Default NVM Value	Description
15:8	Major Revision	0x0	
7:0	Minor Revision	0x0	

6.2.47.45 Reserved Spare Word - 0x0149

Bits	Field Name	Default NVM Value	Description
15:0	Reserved Spare Word	0x0	

6.2.47.46 EMP Image Raw Data - 0x014A

Raw data module length: variable

6.2.47.47 RO Trailer - 0x014B

Raw data module length: variable

6.2.48 1st Free Provisioning Area Section Summary Table

Free area used as the new bank when updating 880 KB long modules.

Word Offset	Description	Reference
0x0000	[New Word]	See section 6.2.48.1

6.2.48.1 [New Word] - 0x0000

Raw data module length: variable

6.2.49 4th Free Provisioning Area Section Summary Table

Free area to be used as the new bank when updating future 64 KB long modules mapped outside the shadow RAM.

Word Offset	Description	Reference
0x0000	[New Word]	See section 6.2.49.1



6.2.49.1 [New Word] - 0x0000

Raw data module length: variable

6.2.50 3rd Free Provisioning Area Section Summary Table

Free area to be used as the new bank when updating future 16 KB long modules mapped outside the shadow RAM.

Word Offset	Description	Reference
0x0000	[New Word]	See section 6.2.50.1

6.2.50.1 [New Word] - 0x0000

Raw data module length: variable

6.2.51 Configuration Metadata - Dummy Section Summary Table

Dummy NVM adaptive configuration metadata.

Section is only included for non-adaptive NVM image.

6.2.52 Configuration MetaData Section Summary Table

NVM adaptive configuration metadata.

Word Offset	Description	Reference



6.2.53 External PHY Global Module Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.53.1
0x0001	External PHY SMBus Master uCode Pointer	See section 6.2.53.2
0x0002	External PHY SerDes uCode Pointer	See section 6.2.53.3
0x0003	External PHY NIMB uCode Pointer	See section 6.2.53.4
0x0004	External PHY Configuration Word 0	See section 6.2.53.5
0x0005	External PHY Configuration Word 1	See section 6.2.53.6
0x0006	External PHY Configuration Word 2	See section 6.2.53.7
0x0007	External PHY Configuration Word 3	See section 6.2.53.8
0x0008	CRC8	See section 6.2.53.9

6.2.53.1 Section Length - 0x0000

Length in words of the section covered by CRC.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.

6.2.53.2 External PHY SMBus Master uCode Pointer - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	SMBus Master uCode Pointer	0xFFFF	

The SMBus Master uCode Pointer points to the External PHY SMBus Master uCode section. For External PHY SMBus Master uCode inner structure detail, see [Section 6.2.54](#).

6.2.53.3 External PHY SerDes uCode Pointer - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	SerDes uCode Pointer	0xFFFF	

The SerDes uCode Pointer points to the External PHY SerDes uCode section. For External PHY SerDes uCode inner structure detail, see [Section 6.2.55](#).

6.2.53.4 External PHY NIMB uCode Pointer - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	NIMB uCode Pointer	0xFFFF	

The NIMB uCode Pointer points to the External PHY NIMB uCode section. For External PHY NIMB uCode inner structure detail, see [Section 6.2.56](#).



6.2.53.5 External PHY Configuration Word 0 - 0x0004

Bits	Field Name	Default NVM Value	Description
15:12	Timeout_10G	0x3	Firmware should set this value in PRTMAC_MACC.ENABLE_TT_BY_TXEN.
11:0	Reserved	0x3	Reserved.

6.2.53.6 External PHY Configuration Word 1 - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x3	Reserved

6.2.53.7 External PHY Configuration Word 2 - 0x0006

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	Reserved
7:0	Reserved	0x5	Reserved

6.2.53.8 External PHY Configuration Word 3 - 0x0007

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x2	Reserved
7:0	Timeout Fixed CTLE	0x1	When the Rx calibration mode is set to the dynamic mode, this is the timeout for the fixed CTLE link attempt. Units of 100 ms.

6.2.53.9 CRC8 - 0x0008

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: External PHY Global Module -> Section Length End Section -> Word: External PHY Global Module -> External PHY Configuration Word 3

6.2.54 External PHY SMBus Master uCode Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.54.1
0x0001	SMBus uCode Version	See section 6.2.54.2
0x0002	SMBus uCode Build ID	See section 6.2.54.3
0x0003	SMBus uCode	See section 6.2.54.4
0x0004	CRC8	See section 6.2.54.5



6.2.54.1 Section Length - 0x0000

Length in words of the section covered by CRC.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.

6.2.54.2 SMBus uCode Version - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Version	0x0	The version of the uCode. This value is compared to the value read from the device to verify the correct uCode download.

6.2.54.3 SMBus uCode Build ID - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Build ID	0x0	The build ID of the uCode. This value is compared to the value read from the device to verify the correct uCode download.

6.2.54.4 SMBus uCode - 0x0003

Raw data module length: variable



6.2.54.5 CRC8 - 0x0004

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: External PHY SMBus Master uCode -> Section Length End Section -> Word: External PHY SMBus Master uCode -> SMBus uCode CRC-8-CCITT: Start Section -> Word: External PHY SMBus Master uCode -> Section Length End Section -> Word: External PHY SMBus Master uCode -> SMBus uCode (last)

6.2.55 External PHY SerDes uCode Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.55.1
0x0001	SerDes uCode Version	See section 6.2.55.2
0x0002	SerDes uCode Build ID	See section 6.2.55.3
0x0003	SerDes uCode	See section 6.2.55.4
0x0004	CRC8	See section 6.2.55.5

6.2.55.1 Section Length - 0x0000

Length in words of the section covered by CRC.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.

6.2.55.2 SerDes uCode Version - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Version	0x0	The version of the uCode. This value is compared to the value read from the device to verify the correct uCode download.

6.2.55.3 SerDes uCode Build ID - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Build ID	0x0	The build ID of the uCode. This value is compared to the value read from the device to verify correct the uCode download.

6.2.55.4 SerDes uCode - 0x0003

Raw data module length: variable



6.2.55.5 CRC8 - 0x0004

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: External PHY SerDes uCode -> Section Length End Section -> Word: External PHY SerDes uCode -> SerDes uCode CRC-8-CCITT: Start Section -> Word: External PHY SMBus Master uCode -> Section Length End Section -> Word: External PHY SMBus Master uCode -> SMBus uCode (last)

6.2.56 External PHY NIMB uCode Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.56.1
0x0001	NIMB uCode Version	See section 6.2.56.2
0x0002	NIMB uCode	See section 6.2.56.3
0x0003	NIMB CSUM	See section 6.2.56.4
0x0004	CRC8	See section 6.2.56.5

6.2.56.1 Section Length - 0x0000

Length in words of the section covered by CRC.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.

6.2.56.2 NIMB uCode Version - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Version	0x0	The version of the uCode. This value is compared to the value read from the device to verify the correct uCode download.

6.2.56.3 NIMB uCode - 0x0002

Raw data module length: variable



6.2.56.4 NIMB CSUM - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	NIMB CSUM	0x0	

6.2.56.5 CRC8 - 0x0004

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: External PHY NIMB uCode -> Section Length End Section -> Word: External PHY NIMB uCode -> NIMB CSUM CRC-8-CCITT: Start Section -> Word: External PHY SMBus Master uCode -> Section Length End Section -> Word: External PHY SMBus Master uCode -> SMBus uCode (last)

6.2.57 5th Free Provisioning Area Section Summary Table

Word Offset	Description	Reference
0x0000	[New Word]	See section 6.2.57.1

6.2.57.1 [New Word] - 0x0000

Raw data module length: variable



6.2.58 X710-TM4 Global Module Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	See section 6.2.58.1
0x0001	Section Length High	See section 6.2.58.2
0x0002	X710-TM4 Configuration Word 0	See section 6.2.58.3
0x0003	X710-TM4 Configuration Word 1	See section 6.2.58.4
0x0004	X710-TM4 Internal PHY Firmware Revision	See section 6.2.58.5
0x0005	X710-TM4 Internal PHY Firmware Date	See section 6.2.58.6
0x0006..(N+6)	X710-TM4 Internal PHY Firmware Code	See section 6.2.58.7
0x0007+N	CRC8	See section 6.2.58.8

6.2.58.1 Section Length - 0x0000

Length in words of the section covered by CRC.

Bits	Field Name	Default NVM Value	Description
15:0	Section Length		Length in words of the section covered by CRC.

6.2.58.2 Section Length High - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Length		Length in words of the section covered by CRC.

6.2.58.3 X710-TM4 Configuration Word 0 - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.

6.2.58.4 X710-TM4 Configuration Word 1 - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	Reserved.



6.2.58.5 X710-TM4 Internal PHY Firmware Revision - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	Version	0x9302	Firmware Revision. Bits 15:12 = FW_REV_BUILD. Bits 11:7 = FW_REV_MAIN. Bits 6:0 = FW_REV_BRANCH. This value is compared to the value read from the device to verify the correct uCode download.

6.2.58.6 X710-TM4 Internal PHY Firmware Date - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Date	0x18E2	Firmware Date. Bits 13,3:0 = Year Bits 12:9 = Month Bits 8:4 = Day This value is compared to the value read from the device to verify the correct uCode download.

6.2.58.7 X710-TM4 Internal PHY Firmware Code - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	Data		This sub-section contains the firmware code for the internal PHY, to be loaded by the X710-TM4 firmware.

Raw data module length: variable

6.2.58.8 CRC8 - 0x0007+N

Bits	Field Name	Default NVM Value	Description
15	CRC Field Used	0x1	0x0 = CRC not used. 0x1 = CRC used.
14:8	Reserved	0x0	Reserved.
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: X710-TM4 Global Module -> Section Length End Section -> Word: X710-TM4 Global Module -> Global Device Configuration Pointer

6.2.58.9 [New Word] - 0x1F6000

Raw data module length: variable



NOTE: *This page intentionally left blank.*



7.0 Shared Resources

7.1 Receive Classification Filters

7.1.1 Introduction

This section describes the receive classification filters that direct inbound packets to a processing engine or cluster out of the LAN engine. The filters also define a specific queue or a context within the selected engine. All the filters described in this section relate to filtering within a specific VSI, which is defined by the switch filters detailed in [Section 7.4.4.3](#).

The classification filters and other header processing units in the device inspect the first 480 bytes of the received packets. If the identified headers exceed 480 bytes, they are reported as an unidentified L2 packet type with no offloads other than Ethernet CRC. They are also handled this way by the classification filters.

7.1.1.1 Association with a Packet Engine

The general rule is that a frame is tested to be LAN.

7.1.1.2 Receive Classification Filters Priority and Usage

Received traffic goes through a set of filters that determine the destination of each received frame. The receive classification filters operate on frames received from the network as well as frames forwarded by the internal switch from a local port (VSI). If a frame is replicated by the internal switch, each replication goes independently through the filters.

The programming interface of the filters are described in the following sections.

These filters operate on received frames listed in priority ordering as shown in [Figure 7-1](#). A frame can match multiple filters. If multiple filters define the same action, then the higher priority enabled matched filter takes precedence. If a frame does not match any of the following filters (with a queue action), the frame is directed to the default queue (queue zero of the VSI).

- Ethertype - see [Section 7.1.6](#)
- Flow Director (FD) - see [Section 7.1.9](#)
- MAC/VLAN - see [Section 7.1.6](#)
- Hash filters (including RSS) - see [Section 7.1.8](#)

Note: In proper programming, the following filters are expected to be mutually exclusive. Meaning, the same packet is not expected to match more than one of them: Ethertype filter and FD filter.

The Hash filter is enabled per function per packet type by the PFQF_HENA for the PFs and the VFQF_HENA for the VFs. The Ethertype filter, MAC/VLAN filter and FD filter are enabled per PF and its VFs by the PFQF_CTL_0 registers.

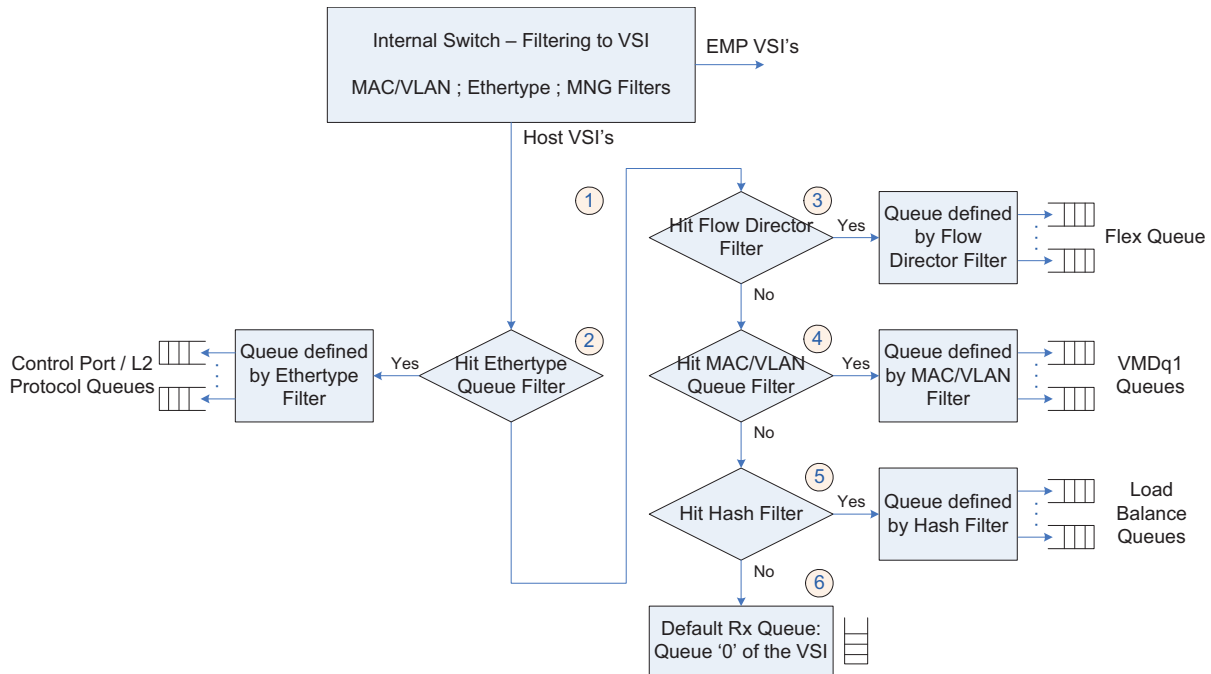


Figure 7-1. Receive Classification Filters - Top Level

Table 7-1 lists the typical use cases of the classification filters and its programming option per function.

Table 7-1. Classification Filters Per PF/VF

VSI	MAC/VLAN, S-Tag and Cloud Switch Filters	Ethertype	Flow Director (FD)	Hash
Main Usage	VMDq1	Control Port / L2 Protocols	Flex Filters ¹	Load Balance
Programming exposed to the PF	Yes	Yes	Yes	Yes
Programming exposed to the VF	No	No	No	Yes

1. There are two main usages for FD filters: software driver controlled filters for ATR and user/operating system controlled filters.

Once a frame matches a filter, the filter provides one or more actions to be performed on the frame. Following are those actions enabled by the filters while Table 7-2 lists which of these actions are enabled per filter type:

- Accept / Reject — A decision whether the frame is dropped by hardware or posted to the device engine for its processing and optionally posted to host memory.
- LAN / A decision whether the frame is posted to LAN engines.
- Receive Queue — Associate the frame to a queue index within the VSI space.
- Increment Statistical Counter — Increment the statistical counters associated with the filter.



- Software field (Filter ID) — Filter ID provided at filter programming and reported back with the matched received packets.
- Packet Bytes — Post bytes from the packet to the receive descriptor. Bytes that should be reported in the receive descriptor can be taken from *Flexible Bytes* in the *Field Vector* (see [Table 7-14](#)).

[Table 7-2](#) lists the type of actions possible by each filter type.

Table 7-2. Filter Actions By Filter Type

Action	MAC / VLAN / Ethertype / S-tag / Cloud Switch Filters	Flow Director	Hash
LAN	LAN	LAN	LAN
Receive queue index / context ID	Queue index within the VSI	Queue index within the VSI	Queue index within the TC range of queues
Accept / reject		Yes	
Increment stat counter (one of 512 GLQF_PCNT)		Yes	
Report filter ID		Filter ID	Hash signature
Report selected bytes from the receive packet		Programmable 4 or 8 bytes	

7.1.1.3 Receive Queue Index

As indicated in the previous section, one of the filter’s action is the packet classification to LAN queues. The LAN queue index defined by the classification filters is a relative index within a specific VSI that is defined by the embedded switch. Assuming a classification filters defines a queue index ‘n’ then:

- VSI that is defined by the VSILAN_QTABLE (VSILAN_QBASE.VSIQTABLE_ENA = 1b). The queue index in the PF space equals to: VSILAN_QTABLE[n/2].QINDEX_0 for even ‘n’ and VSILAN_QTABLE[(n-1)/2].QINDEX_1 for odd ‘n’.
- VSI that is defined by the VSILAN_QBASE (VSILAN_QBASE.VSIQTABLE_ENA = 0b. The queue index in the PF space equals to VSILAN_QBASE.VSIBASE + ‘n’.
- VF queues: The mapping of the previous queue indexes to relative indexes in the VF space is not a direct mapping. The mapping is defined by the queue indexes in the PF space that are programmed in the VSILAN_QTABLE and the VPLAN_QTABLE registers. For example, assuming a VF with 4 queues in a single VSI with the following settings: VPLAN_QTABLE[0:3]=A,B,C,D (while A...D are the queue indexes in the PF space); VSILAN_QTABLE[0].QINDEX_0,QINDEX_1 = C,D and VSILAN_QTABLE[1].QINDEX_0,QINDEX_1=A,B. Then receiving a packet to queue ‘0’ of the VSI is mapped to queue C in the PF space which is queue index 2 in the VF space.

The hash filter is more complex, while the queue index is a relative index within a specific range of queues (usually related to a TC) of a VSI. See [Section 7.1.8](#).

Following are some exception rules for the receive queue index:

- If a frame does not match any of the following classification filters (with a queue action), the frame is directed to the default queue (queue zero of the VSI).
- If the packet has a MAC error (reflected by the RXE flag in the receive descriptor), the packet is directed to the default queue (queue zero of the VSI that is defined by the PRT_SBPVSI register). Note that a packet with MAC error is posted to the host only if enabled by the SBP flag in the PRT_SBPVSI register.
- The receive queue index that defines an invalid queue causes dropping of the matched received packets. Such packets are counted by the GLV_REPC counter of the VSI.



- The following sub-bullets list the cases that a queue index is considered invalid:
 - A VSI that is defined by VSILAN_QTABLE is limited to 16 queues, which is the size of this table. Therefore, a queue index that equal to or larger than 16 is considered invalid.
 - The queues in a VSI that is defined by VSILAN_QTABLE are enabled by the QINDEX_0 and QINDEX_1 parameters. A queue index that points to a VSILAN_QTABLE entry with a value of 0x7FF is considered invalid.
 - A VSI that is defined by VSILAN_QBASE.VSIBASE is limited only by the PF queue space. A queue index that exceeds the PF space is considered invalid.

7.1.1.4 Initialization

This section lists all registers associated with the classification filters directly or indirectly and its required initialization. Note that some of the text in this section might not be clear, lacking a complete description of the registers. This description is given in the following sections as well as in the Programming Interface section. The initialization sequence should be executed according to the following order (described in the sub-sections that follow):

- Function Private Memory Allocation
- Queue Allocation
- Static Classification Filters Registers
- Dynamic Classification Filters Registers

7.1.1.4.1 Function Private Memory Allocation

The LAN filters are stored in the Function Private Memory (FPM) while part of them are fetched to the on-chip cache. The FPM allocation registers are described in the Function Private Memory Space Configuration section. These registers should be initialized per PF prior to any settings of the LAN queues.

7.1.1.4.2 Queue Allocation

LAN queues should be allocated to the functions before packets are directed to these queues. If this is not done, packets are dropped.

7.1.1.4.3 Static Classification Filters Registers

Table 7-3. Static Classification Filters Registers Initialization

Register	Initialization Comments
PRTQF_CTL_0	Symmetric hash enablement is loaded from the NVM.
GLQF_CTL	Global classification parameters are loaded from the NVM.
PFQF_CTL_0	PF classification parameters should be initialized before LAN contexts programming.
VPQF_CTL	LAN filter parameters should be initialized after the PFQF_CTL_0 register of the PF and before LAN contexts programming.
GLQF_HSYM	Swap register and symmetric hash register loaded from the NVM.



Table 7-3. Static Classification Filters Registers Initialization

Register	Initialization Comments
GLQF_NETKEY_PIT	The UDP network key structure is programmed by the Set Network Key Structure admin commands before any of the Add Tunneling UDP admin command are initiated.
PRTQF_FD_MSK	Mask registers for the input set of the flexible payload for the FD filter is loaded from the NVM or programmed by the PFs at software device driver initialization time.
VSIQF_TCREGION	The queue regions for 8 x TCs are programmed by the Add/Update VSI admin commands.
PFQF_FDALLOC	FD filter table allocation to PFs is loaded from the NVM.

7.1.1.4.4 Dynamic Classification Filters Registers

Table 7-4. Dynamic Classification Filters Registers Initialization

Register	Initialization Comments
PFQF_HENA and VFQF_HENA	PCTYPE enablement for the hash filters are programmed by the function (PF or VF) at run time.
PFQF_HREGION and VFQF_HREGION	Override the TC queue regions for specific PCTYPES registers are programmed by the function (PF or VF) at run time.
PFQF_HKEY, PFQF_HLUT, VFQF_HKEY and VFQF_HLUT	The hash key and LUT registers are programmed by the function (PF or VF) at run time.

7.1.2 Packet Types and Input Set

The X710-TM4 parses the first 480 bytes of the received packets. Hardware identifies the protocol layers upon which packet classification types (PCTYPE’s) are defined. The supported PCTYPE’s are listed in Table 7-5. The fields used for the classification filters (named as Input Set) are defined per filter per PCTYPE as listed in the Table 7-5. The table contains a shared column for the Input Set of the FD filter. The following section describes which PCTYPES are relevant to each filter.

Note: L2 packet types are identified only for the last EThertype in the outer L2 header.

Table 7-5. Packet Classifier Types and its Input Sets

PCTYPE	PCTYPE Description	Hash Input Set	FD Input Set
Reserved			
0 ... 25	Reserved for future use	-	-
Tunneling			
26	Geneve OAM	Source outer UDP Port, VNI	Source outer UDP Port, VNI
27	VXLAN-GPE OAM	Source outer UDP Port, VNI	Source outer UDP Port, VNI
[Inner] IPv4 Packets			
28	Reserved		
29	Reserved		
30	Reserved		
31	NonFIPv4, UDP	IP4-S, IP4-D, UDP-S, UDP-D	IP4-S, IP4-D, UDP-S, UDP-D
32	Reserved		



Table 7-5. Packet Classifier Types and its Input Sets

PCTYPE	PCTYPE Description	Hash Input Set	FD Input Set
33	NonFIPv4, TCP	IP4-S, IP4-D, TCP-S, TCP-D	IP4-S, IP4-D, TCP-S, TCP-D
34	NonFIPv4, SCTP	IP4-S, IP4-D, SCTP-S, SCTP-D, SCTP-Verification-Tag	IP4-S, IP4-D, SCTP-S, SCTP-D, SCTP-Verification-Tag
35	NonFIPv4, Other	IP4-S, IP4-D	IP4-S, IP4-D
36	FragIPv4	IP4-S, IP4-D	IP4-S, IP4-D
37	Reserved	-	-
[Inner] IPv6 Packets			
38	Reserved		
39	Reserved		
40	Reserved		
41	NonFIPv6, UDP	IP6-S, IP6-D, UDP-S, UDP-D	IP6-S, IP6-D, UDP-S, UDP-D
42	Reserved		
43	NonFIPv6, TCP	IP6-S, IP6-D, TCP-S, TCP-D	IP6-S, IP6-D, TCP-S, TCP-D
44	NonFIPv6, SCTP	IP6-S, IP6-D, SCTP-S, SCTP-D, SCTP-Verification-Tag	IP6-S, IP6-D, SCTP-S, SCTP-D, SCTP-Verification-Tag
45	NonFIPv6, Other	IP6-S, IP6-D	IP6-S, IP6-D
46	FragIPv6	IP6-S, IP6-D	IP6-S, IP6-D
47. . .51	Reserved	-	-
L2 Packet Types			
52 ... 62	Reserved for future use	-	-
63	L2 packet type (1)	L2 Ethertype	L2 Ethertype

7.1.3 Filter’s Input Set

The input set is defined per filter type as follows:

- Layer 2 filters that are used by the switch logic. These include: MAC filters; VLAN filters; S-tag and Ethertype. The input set for these filters is defined in the Internal Switch section.
- The input set for the FD filter is flexible. Its input set for the 64 global PCTYPES has two components:
 - A pre-defined byte stream as listed in [Table 7-5](#) (up to total of 24 x 2 byte words programmed by the PRTQF_FD_INSET loaded from the NVM).
 - A flexible input set defined by the PRTQF_FD_FLXINSET registers. A PRTQF_FD_FLXINSET register ‘n’ matches PCTYPE ‘n’ while each bit ‘i’ in the 8-bit *Inset* field in these registers enable word ‘i’ in the flexible field vector (mapped to bit 28+‘i’ in the PRTQF_FD_INSET register of the matched PCTYPE). If only a portion of a 16-bit word is required for the input set, the non-relevant bits in the flexible input set can be masked out by the PRTQF_FD_MSK registers.
- The input set for the hash filter for the 64 global PCTYPES is listed in [Table 7-5](#) (up to total of 24 x 2 byte words programmed by the GLQF_HASH_INSET registers that are loaded from the NVM).



7.1.4 Flexible Payload

The X710-TM4 supports a flexible payload of 8 words (16 bytes) that are extracted from the payload of the packet. The 8 words can be extracted from up to 3 offsets within the payload. Payload is defined as follows:

- If packet is identified as a L2 packet, payload is anything after the last EtherType.

Else,

- If packet is identified as a L3 packet, payload is anything after the inner identified L3 header.

Else,

- If packet is identified as a L4 packet, payload is anything after the inner identified L4 header.

The flexible payload is stored in a field vector. This vector can be used by the FD filter as described in the following sections.

7.1.5 Protocol Layers Filter’s Fields and Packet Classifier Types

The X710-TM4 parses the first 480 bytes of the received packets. Hardware identifies the protocol layers in this range and constructs a field vector used for all the filters. Each filter uses a different input set taken from the shared field vector. The protocol layers, field vector and filter fields as well as the input set are defined in the following subsections. The fields extracted from the protocol layers to the field vector are pre-defined.

7.1.5.1 Protocol Layers (Type Rule Bus)

Protocol layers are portions of the packet that are associated with specific protocols as listed in [Table 7-6](#). The protocol layers table includes L2 layers (like LLDP), L3 protocols (like IP header), L3 protocols (like TCP header) and extracted flags from the packet (like IP fragmentation indication and TCP URG). The protocol layers and the flags are used to construct the field vector and define packet types as explained in the proceeding sections.

Table 7-6. Protocol Layers Table (Type Rule Bus)

Index	Protocol Layer	Index	Protocol Layer
0	L2 Payload for ETYPE = 0x88F7 ¹	32	Inner Cloud MAC Without VLAN
1	L2 Payload for ETYPE = 0x8914 (FIP Packet) ¹	33	L2 Payload
2	VXLAN / VXLAN-GPE Key (Key 0)	34	L3 Payload
3	Geneve Key (Key 1)	35	L4 Payload
4	L2 Payload for ETYPE = 0x88CC (LLDP) ¹	36	L2 Header 0 - PreL2 Header
5	L2 Payload for ETYPE = 0x8847 MPLS Header ¹	37	L2 Header 1 - MAC Addresses
6	Place Holder for Generic UDP Tunneling (key 2)	38	L2 Header 2 - Reserved
7	NSH ETYPE = 0x894F	39	L2 Header 3 - S-Tag

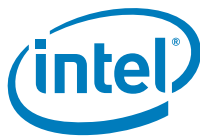


Table 7-6. Protocol Layers Table (Type Rule Bus)

Index	Protocol Layer	Indx	Protocol Layer
8	L2 Payload for ETYPE = 0x888E (IEEE 802.1X) ¹	40	L2 Header 4 - External VLAN (In the outer MAC header in case of MAC in UDP or MAC in GRE packets)
9	L2 Payload for ETYPE = 0x0806 (ARP) ¹	41	L2 Header 5 - VLAN (Inner one in case of double VLANs. Both are in the outer MAC header in case of MAC in UDP or MAC in GRE packets)
10	GRE Key ¹	42	L2 Header 6 - CTS
11	Reserved	43	L2 Header 7 -
12	IPv4 Layer (Inner IPv4 in case of tunneling)	44	L2 Header 8 - Reserved
13	IPv6 Layer (Inner IPv6 in case of tunneling)	45	L2 Header 9 - Reserved
14	Reserved	46	Reserved
15	GRE no Key ¹	47	Reserved
16	Teredo Protocol Layer	48	IP Header is Fragmented
17	TCP Protocol Layer	49	Inner IPv4 Header Includes IP Options
18	UDP Protocol Layer	50	TCP Header Includes Options
19	SCTP Protocol Layer	51	Reserved
20	ICMP Protocol Layer	52	External IPv4 Layer
21	Reserved	53	External IPv6 Layer
22	Reserved	54	Reserved
23	ICMPv6 Protocol Layer	55	TCP SYN Flag
24	IPv6 extension protocol layer(s) other than destination/routing (in case of tunneling it is an OR function of the inner and outer headers)	56	Reserved
25	IPv6 Extensions destination/routing (in case of tunneling it is an OR function of the inner and outer headers)	57	OAM Flag in VxLAN-GPE Header
26	FC RDY or RD/WR Request (R_CTL = 0x05 or 0x06)	58	OAM Flag in Geneve Header
27	FC Data (_RCTL = 0x01)	59	Reserved for Flexible Flag
28	FC VFT Header	60	Reserved for Flexible Flag
29	FC RSP (R_CTL = 0x07)	61	Unicast Destination MAC Address Indication
30	FC Other	62	Multicast Destination MAC Address Indication
31	Inner Cloud MAC + VLAN	63	Broadcast Destination MAC Address Indication (multiplexed with last Ethertype internally)

1. For tunneled packets, these flags indicates the existence of matched protocol layers in the outer L2 header.

7.1.5.2 Filter Field

Filter field is a consecutive byte stream extracted from a protocol layer to an internal field vector that might be used for the packet classification. A field size can be up to 30 bytes defined in 2-byte granularity.



7.1.5.3 Field Vector

The field vector is a 128-byte string composed of multiple fields that are extracted from identified protocol layers of the packet. The field vector is initialized to zero before it is filled by received packet fields. Out of the 128 bytes of the field vector only the first 116 bytes are available for the packet's fields while the other bytes are reserved for internal parameters. Among these parameters are the function ID, VSI index, identified protocol layers and protocol flags. [Table 7-7](#) lists the fields that are extracted from the packets (according to the device setting in the default NVM image). The fields are posted to the field vector in a mixed big/little endian ordering while the first byte on the wire is posted in the MS byte of the first word number of the field. For example, the first byte on the wire of the destination MAC address (the MS byte) is posted to high byte of word number 0 in the field vector. Note that mutual exclusive protocol layers can share the same space in the field vector (like TCP and UDP).

In general, mutually exclusive protocol layers can share the same words in the field vector (like TCP and UDP). In some special cases it might make sense to share the same words in the filter for non mutually exclusive protocol layers. In such a case, the most recent detected protocol layer would step over previous protocols layer's words that are defined in the same location in the field vector.

Table 7-7. Default Field Vector Table

Word Num	Protocol Layers				
	L2 Protocol Layers				
0:2	Destination MAC Address (in outer or single L2 header)				
3:5	Source MAC Address (in outer or single L2 header)				
6	Reserved	S-tag	Reserved	VLAN Tag (in the inner L2 header) see Table 7-8	
7		-			
8	Inner or Single VLAN Tag (in outer or single L2 header) see Table 7-8				
	L3 Protocol Layers				
	Inner or Single IPv4	Inner or Single IPv6		ARP	
9	First 8 words of the IPv4 header (up to including the source IP address)	First 4 words of the IPv6 header (up to including the hop limit)		Hardware type ... Sender IP address (first 28 bytes of the header)	
10					
11:12			Complete FC Header		
13:16			IPv6 Source Address		
17:20	0x00				
21:22	0x00	IPv6 Destination Address			
23:26	0x00		0x00	0x00	
27:28	Destination IP Address		0x00	0x00	
	L4 Protocol Layers				
	TCP	UDP	SCTP	ICMPv6	
29:30	First 16 bytes of the TCP header	First 8 bytes of the UDP header	First 8 bytes of the SCTP header	Words 0, 1 of the header	
31:32				0x00	
33:36				0x00	0x00
37:38				0x00	0x00
	Tunneling Layers and Flexible Payload + Last EtherType				
	Non-tunneled	IP in IP Tunneling	UDP Tunneling	GRE Tunneling	



Table 7-7. Default Field Vector Table

39:41	0x00	0x00	Destination MAC Address (in the inner L2 header). Relevant only for MAC in UDP or MAC in GRE.	
42:43	0x00	0x00	Tunneling UDP Source and Destination Port Numbers	GRE Version and Protocol Type
44:45	0x00	0x00	Tunneling UDP Keys	GRE Key
46	0x00	0x00	Geneve / VXLAN Version (word 0 of the header)	
47:48	0x00	0x00	Tunneling UDP Keys	0x00
49	Last EtherType (in case of tunneled packets it is the last Ethertype in the outer L2 header)			
50:57	Outer or single destination IP address Note that if this field is required then the flexible payload must not be enabled or else the flexible payload will override this field.			
Other Internal Fields				
58:63				

Table 7-8. Detailed L2 Tags in the Field Vector (Assuming all L2 tags are VLANs)

	Word 7: VLAN Tag (in the inner L2 header)	Word 8: Inner or Single VLAN Tag (in outer or single L2 header)
Non-tunneled received packet with a single VLAN.	X	VLAN.
Non-tunneled receive packet with double VLANs.	X	Inner VLAN.
Tunneled received packet with a single VLAN in the outer L2 header and no VLAN in the inner L2 header.	X	VLAN in the outer L2 header.
Tunneled received packet with double VLAN in the outer L2 header and no VLAN in the inner L2 header.	X	Inner VLAN in the outer L2 header.
Tunneled received packet with a single VLAN in the outer L2 header and VLAN in the inner L2 header.	VLAN in the inner L2 header.	VLAN in the outer L2 header.
Tunneled received packet with double VLAN in the outer L2 header and VLAN in the inner L2 header.	VLAN in the inner L2 header.	Inner VLAN in the outer L2 header.

7.1.6 Layer 2 Classification Filter

The X710-TM4 supports L2 filters (MAC / VLAN / L2 EtherType) in the embedded switch. These filters act on the following layers: S-tag; MAC address; VLAN and EtherType. These filters define a VSI and can also define a unique LAN queue within that VSI. The LAN queue is enabled by the *ToQueue* flag and the queue is defined by the queue number for each filter. These parameters are programmed by admin commands described in [Section 7.4.9.5.9](#). The filters can be defined as the second or fifth priority filters for receive queue classification as follows:

Priority # Section 7.1.1.2	Filters Defined by the Following Admin Command(s)
2	Add control packet filter.
5	Add MAC, VLAN pair, add S-tag and add cloud filters.



The Ethertype filter and MAC/VLAN filter affect on queue routing are enabled per PF and its VFs by the PFQF_CTL_0 registers.

7.1.7 Tunneled Packets

The X710-TM4 supports tunneled packet formats including tunneled IP address, UDP Teredo, MAC in UDP, IP in GRE and MAC in GRE. These tunneling packets are supported for transmit and receive data processing offloads. When it comes to packet classification, it is based only on the encapsulated packet (exactly the same as non-tunneled packets). Following are the supported tunneled packets:

IP Tunneling

For a non-tunneled IP packet, there is only one IP header that is used for packet classification. In case of tunneling, The inner IP header is used for packet classification while the outer IP header is used for the switch filters.

Teredo

Teredo are supported protocol layers for classification filtering. See [Section 7.1.7.1](#) for more detail. Teredo support is mutually exclusive with MAC in UDP and GRE tunneling.

MAC in UDP

MAC in UDP packet format is described in [Section 8.2.3](#). MAC in UDP packet is identified by the destination UDP port number. The UDP ports used for MAC in UDP are programmed by the same admin commands as the Teredo UDP ports as described in [Section 7.1.7.1](#). Classification to receive queues is based on the encapsulated packet according to its structure. Filtering to VSI for MAC in UDP packets is described in [Section 7.4.4.4.3](#). MAC in UDP support is mutually exclusive with Teredo and GRE tunneling.

GRE Tunneling

IP in GRE and MAC in GRE packet formats are described in [Section 8.2.3](#). GRE tunneling is identified by the last Ethertype in the outer L2 header. IP in GRE or MAC in GRE are identified by the *Protocol Type* field in the GRE header. Filtering to VSI for MAC in UDP packets is described in [Section 7.4.4.4.3](#). MAC in UDP support is mutually exclusive with Teredo and MAC in UDP.

The previous text states that packets are classified to receive queues based on the encapsulated packet fields. Packets with a Geneve header or VXLAN-GPE with an active *OAM* flag are the exception to this rule. These packets are identified as unique PCTYPES as listed in [Table 7-5](#). Such packets with an active *OAM* flag can be directed to receive queue zero of the VSI (default queue) if programmed by software. In order to get this functionality, The X710-TM4 should not process these packets by the hash and FD filters. Software can do so by disabling the OAM PCTYPES in the xxQF_HENA registers of the function (affecting the hash filter).

7.1.7.1 Unique UDP Port Filters

There are some destination UDP port numbers that have unique meaning. These ports include 1588 packet identification, Teredo tunneling. X710-TM4 supports the following port numbers:

- Two global port numbers for 1588 equal to 319 and 320.
- A tunneling UDP table with 16 global port numbers for Teredo tunneling or MAC in UDP tunneling that are programmed at run time by admin commands are described in the following sections. The tunneling UDP port numbers should not be one of the values reserved for 1588 and should not be set to 0x0000 or 0xFFFF.



Note that the X710-TM4 does not process (offload) the checksum field in the tunneling UDP header.

The tunneling UDP table is a global resource available for all PFs on a first come first served policy. This table defines the port number and the tunneling type listed in [Table 7-10](#). The PFs can add an entry or delete an entry by admin commands. Any add or remove command is acknowledged by a completion command that provides a status of the requested action and an updated status of the table resources.

A table entry is added by the first PF that programs it. Any additional PF that programs the same UDP port number just registers to the existing entry. Once a table entry is programmed, it affects received packets for all PFs on all LAN ports (regardless which PFs are registered to that UDP port number).

The admin commands used to program the tunneling headers are listed in [Table 7-9](#) and detailed in the following sections:

Table 7-9. Tunneling UDP Admin Commands

Name	Opcode	Ref	Admin Command Type
Add Tunneling UDP	0x0B00	Section 7.1.11.1	Direct
Remove Tunneling UDP	0x0B01	Section 7.1.11.2	Direct

Table 7-10. Supported Tunneling Types

Tunneling Type	Tunneling UDP Protocol Type	Fixed Header Length [bytes]	Variable Header Length	Network Key
Teredo	0x10 (next protocol is IP)	0x8	No	No
VXLAN	0x00 (next protocol is MAC)	0x10	No	Yes, see Section 7.1.7.1.1
Geneve	0x01 (next protocol is defined by the Geneve header)	0x10	Defined by the <i>Options Length</i> field in the Geneve header	Yes, see Section 7.1.7.1.1
Place holder for Generic MAC in UDP	0x02 (next protocol is MAC)	Not defined yet	No	Yes, see Section 7.1.7.1.1
Reserved	Else	N/A	N/A	N/A

7.1.7.1.1 Network Key Structure of Tunneling Header

There are four global network keys, shared for all LAN ports. The network keys are used to identify a cloud tenant. The X710-TM4 supports a programmable structure for these network keys that can be programmed by the PFs, using the Program Network Key Structure admin command. In NVGRE and Geneve headers, it is defined as a 3-byte field at a pre-defined offset in the header. The matched Network Key Structure parameters for the supported tunneling headers are listed in the following table.

Key Index	Tunneling Header	Key Offset	Key Length
0	VXLAN / VXLAN-GPE	12	3
1	Geneve	12	3
2	Place holder for generic MAC in UDP	Offset 1: 8 <= Offset 1 < 32 Offset 2: 8 + Offset 1 + Length 1 <= Offset 2 < 64 - Length 2	0 <= Length 1, Length 2 <= 6 (*) Length 1 + Length 2 <= 10 (*) (*) Assuming word aligned keys
3	NVGRE	4	3



7.1.8 Hash Filter

The hash filter is a mechanism to statistically distribute received packets into several receive queues. Software allocates the queues among the different processors, therefore sharing the load of packet processing among several processors. One of the most common use cases of hash filters is the RSS hash defined by Microsoft* and used widely by other operating systems as well. Other generic hash functions are also used by embedded systems. The hash filter directs the received packets to queue index within a region of queues of the VSI as shown in the [Figure 7-2](#). Following are some terms relating the hash filter.

Packet Classifier Type	The PCTYPE is the lowest index PCTYPE that is enabled for the function. The PCTYPE's are enabled for the hash filter by the PFQF_HENA registers for the PFs and by the VFQF_HENA registers for the VFs. See Table 7-5 for the PCTYPE's and its priority order.
Input Set	The input set for the hash filter are defined globally per PCTYPE by the GLQF_HASH_INSET registers as listed in Table 7-5 .
Hash Function	The 32-bit hash function is based on a Toeplitz algorithm or simple XOR scheme as described in Section 7.1.10 . The hash can be calculated on the packet's fields as is or can be symmetric hash as described in Section 7.1.10.3 . The hash signature is reported in the receive descriptor if the space is not taken by other filters as described in Section 8.3.2.2 . The hash is calculated only on the input set words. Masked bits within the input set words (defined by GLQF_HASH_MSK registers) are replaced by zero's for the hash calculation.
Queue Index LUT	The queue index LUT gets the LS bits of the hash output and provides a queue index within a region. The LUT in each PF gets the 9 LS bits of the hash output having either 128 or 512 entries (as defined by the <i>HASHLUTSIZE</i> field in the PFQF_CTL_0 register). Each entry in the LUT defines receive queues in the range of 0 to 63. The LUT in each VF gets the 6 LS bits of the hash output having 64 entries while each entry defines receive queues in the range of 0 to 15. These LUTs are configured by the PFQF_HLUT and VFQF_HLUT registers for the PFs and VFs, respectively.
Receive Queue Regions	The VSIs support 8 regions of receive queues that are aimed mainly for the TCs. The TC regions are defined per VSI by the VSIQF_TCREGION register. The region sizes (defined by the TC_SIZE fields) can be any of the following value: 1, 2, 4, 8, 16, 32, 64 as long as the total number of queues do not exceed the VSI allocation. These regions start at the offset defined by the TC_OFFSET parameter. According to the region size, the 'n' LS bits of the queue index from the LUT are enabled. The hash filter defines the queue index within the queue region of the VSI. The queue region is defined by the traffic class or the PCTYPE as programmed by the OVERRIDE_ENA and REGION fields in the PFQF_HREGION registers for the PFs and VFQF_HREGION registers for the VFs (as illustrated in Figure 7-2).
VSI Queue Index	The outcome VSI queue index is shown by the following expressions: $LUT_OUT + VSIQF_TCREGION.TC_OFFSET$, while... $LUT_OUT = (VSIQF_TCREGION \rightarrow TC_SIZE)$ LS bits of the QF_LUT $QF_LUT = PFQF_HLUT[9 \text{ LS bits of the packet hash}]$ for the PF or $VFQF_HLUT[6 \text{ LS bits of the packet hash}]$ for the VF

Outcome Queue Index

Mapping the VSI queue index to the PF index space is done by VSILAN_QTABLE or VSILAN_QBASE as explained in Section 8.2.1.2.

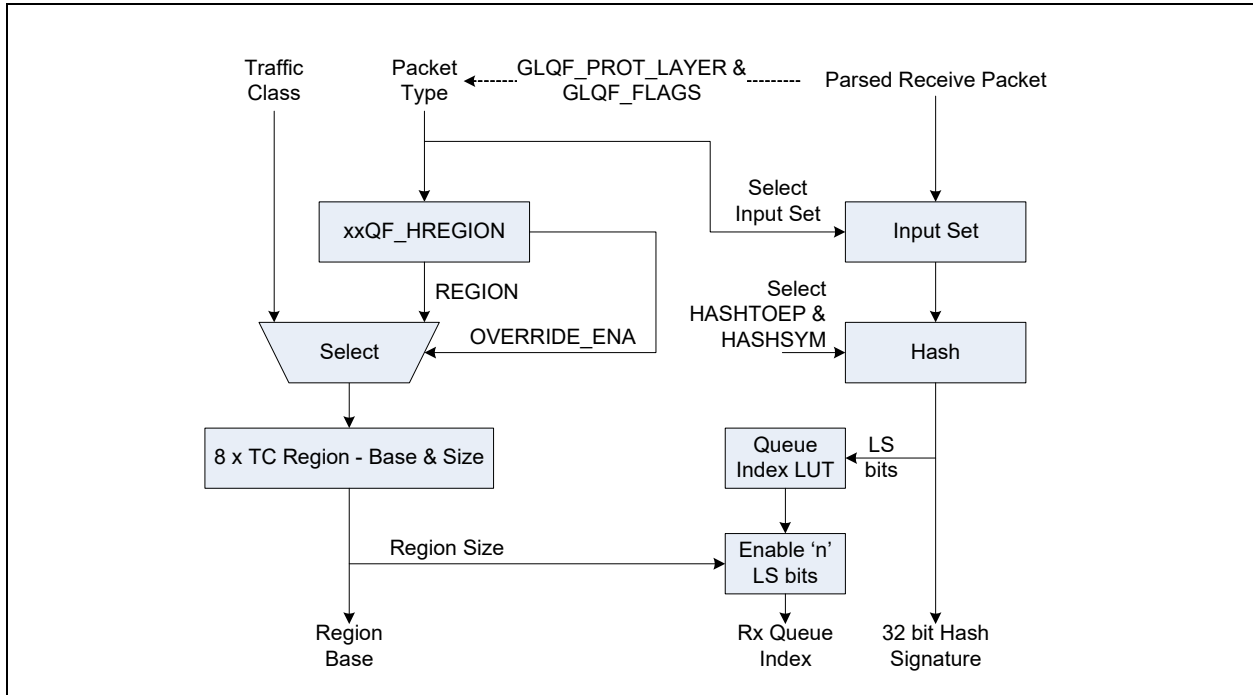


Figure 7-2. RSS Block Diagram

7.1.8.1 Comments for the Init Section

Dynamic setting options (expected at run time):

- VSIQF_TCREGION registers per VSI that define the hash region sizes (should be dynamic due to the dynamic LQP allocation)
- PFQF_HLUT registers and VFQF_HLUT registers per function.

7.1.9 FD Filter

The FD filter is aimed to match specific flow or flows with some action. The FD filter is based on an exact match of the selected tuples named as input set for filtering purposes. The FD filter is composed of the following components.

Filter Enable Option

The FD filtering is enabled by the following parameters. Filtering is enabled per PF and its VFs by the PFQF_CTL_0 registers.



Programming

FD filter programming is done by the Flow Director Programming Descriptor described in [Section 8.4.2.3](#) followed by packet structure that contains the filter fields as shown in [Figure 7-3](#). Failed programming and removal of filter entries are reported by the Programming Status Descriptor described in [Section 8.3.2.2.3](#). The packet that is used to program the filter can be optionally transmitted depending on the dummy flag in the transmit data descriptor. Note also that the packet used for the programming can be a single packet or part of a TSO. Removing a single filter is done by the same programming descriptor.

Filter programming is possible only from transmit queues that are enabled by the *FDENA* flag in the transmit queue context. Setting the *FDENA* flag is expected only for the PF queues. The PF can program a filter for any of its own VSIs or to VSIs that belong to its VFs. The target VSI for the matched packets is defined by the *DEST_VSI* parameter in the Filter Programming Descriptor. The PF is permitted to program a FD filter only if the FD filter is enabled by the *FD_ENA* flag in the *PFQF_CTL_0* register. When the *PCTYPE* in the FD Programming Descriptor is not used (*FD_AUTO_PCTYPE* flag in the *GLQF_CTL* is set), software is unaware of the *PCTYPE*.

The *Source* and *Destination* fields in the transmitted packet are presented in a reversed order with respect to the expected received packets. During filter programming time, hardware swaps these fields. Attempt to re-program an existing filter entry updates the filters parameters.

At programming time, the filter can be added at the expense of the guaranteed space or the best effort space of the function as described in [Section 7.1.9.1](#). In those cases, it is required to know up front that a filter programming is accepted by hardware and software should track its guaranteed space by reading *GUARANT_CNT* in the *PFQF_FDSTAT* register.

In some cases, software might need to clear the FD table. Setting the *CLEARFDTABLE* flag in the *PFQF_CTL_1* register, all entries of the PF are invalidated from the FD table. This action can be done during normal transmission and reception. This action might take some time. Software should poll the FD filter counters of the PF in the *PFQF_FDSTAT* register until it is cleared as an indication that the clear all PF entries sequence completed.

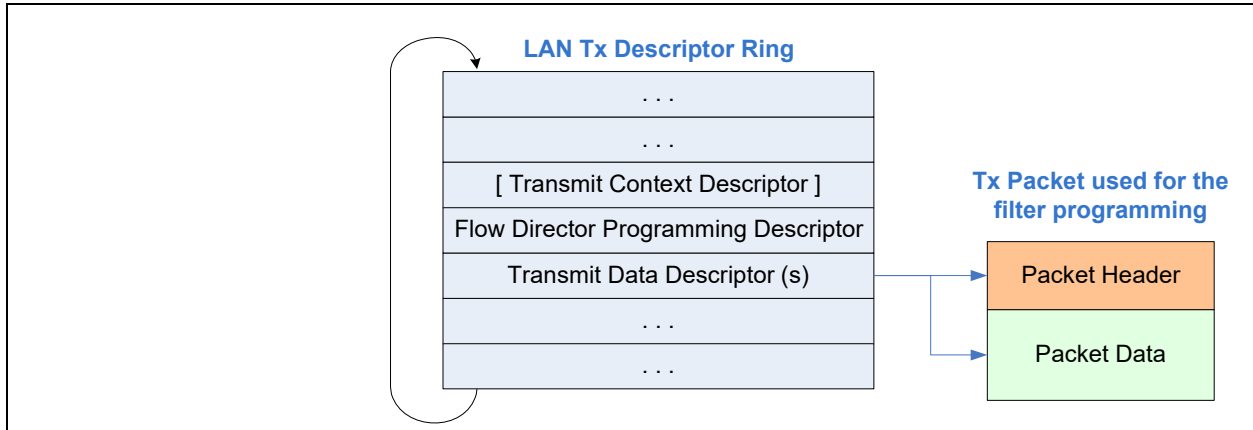


Figure 7-3. Flow Director Filter Programming

Packet Classifier Type

All PCTYPE's described listed in [Table 7-5](#) are supported for the FD filter.

Input Sets

As opposed to all other filters, the input sets for the FD filter are defined per port (by the PRTQF_FD_INSET registers) enabling different input set for the flexible payload layers (by the PRTQF_FD_FLXINSET registers). The masking option for the standard protocol layers are defined globally by the GLQF_FD_MSK registers. These registers are loaded from the NVM and should not be modified by software. The masking options for the flexible payload protocol layers are defined per port by the PRTQF_FD_MSK registers. These registers can be loaded from the NVM or programmed by software at init time. [Table 7-5](#) lists the default setting of the PRTQF_FD_INSET and GLQF_FD_MSK registers that address the standard protocol layers.

FD Filter Entry Context

Each filter entry consists of 64 bytes. As previously indicated, the input set can be defined as large as 48 bytes while the other 16 bytes are reserved for filter action, table management parameters and fields that are compared against the received packet:

- The PCTYPE and the DEST_VSI that are provided in the FD filter programming descriptor.
- The programming PF index.
- The QINDEX, FLEXOFF, STAT_CNT, FDID, DEST and FD_STATUS.



Filter Match Criteria

A packet matches a filter entry if the following conditions are met:

1. The target VSI equals to the programmed DEST_VSI.
2. The identified PCTYPE equals to the programmed PCTYPE.
3. The received packet pattern matches the programmed one (the relevant fields for the PCTYPE as defined by the FD input set listed in [Table 7-5](#)).

Filter Action

The filter action is programmed as part of the filter programming command. The actions enabled for the FD filter are listed in [Table 7-2](#). The filter action is taken when the packet matches the filter entry.

FD Table Size

The X710-TM4 supports 8K x 64 byte filters. Out of these bytes, only 48 bytes can be used for the input set.

Hash and Buckets

The FD table is organized in buckets. The number of buckets is equal to 16K (twice the number of supported filters). Each bucket might be: empty, include a single filter entry or multiple filter entries. These buckets are managed autonomously by hardware.

7.1.9.1 FD Table Allocation

The FD table is a shared resource for all functions. The allocation between the PFs is loaded from the NVM to the PFQF_FDALLOC registers per PF. The FD supports two kinds of allocated spaces per PF: guaranteed space and best effort space. Each PF gets a private space which is a guaranteed number of entries in the FD table (defined by the FDALLOC parameter). The sum of the guaranteed spaces of all PFs must be smaller than or equal to the size of the FD table. If the sum of the FDALLOC parameters is smaller than the size of the FD table, the remainder of the space can be used by all functions as best effort. The FDBEST parameter in the PFQF_FDALLOC registers define the permitted use of the best effort space by each PF. The total size of the best effort space is defined by a global FDBEST in the GLQF_CTL register (loaded from the NVM). This parameter must not exceed the size of the FD table minus the sum of the FDALLOC parameters of all PFs.

Note: The maximum value of the global FDBEST parameter in the GLQF_CTL register can be set to 8K minus 32. In order to make use of all 8K filter entries, at least one of the PF's FDALLOC parameters should be larger than zero.

At filter programming, hardware tries both spaces: guaranteed and best effort. Priority between usage of these spaces is set by the *PROGPRIOR* flag in the global GLQF_CTL register. At filter invalidation, the budget is gained back to either the guaranteed space or the best effort space according to a priority setting by the *INVALPRIOR* flag in the global GLQF_CTL register.

Software can track the FD table population (best effort space as well as the guaranteed space). GLQF_FDCNT_0 reports the total number of filter entries in the entire FD table while PFQF_FDSTAT reports the number of filter entries used by the PF.

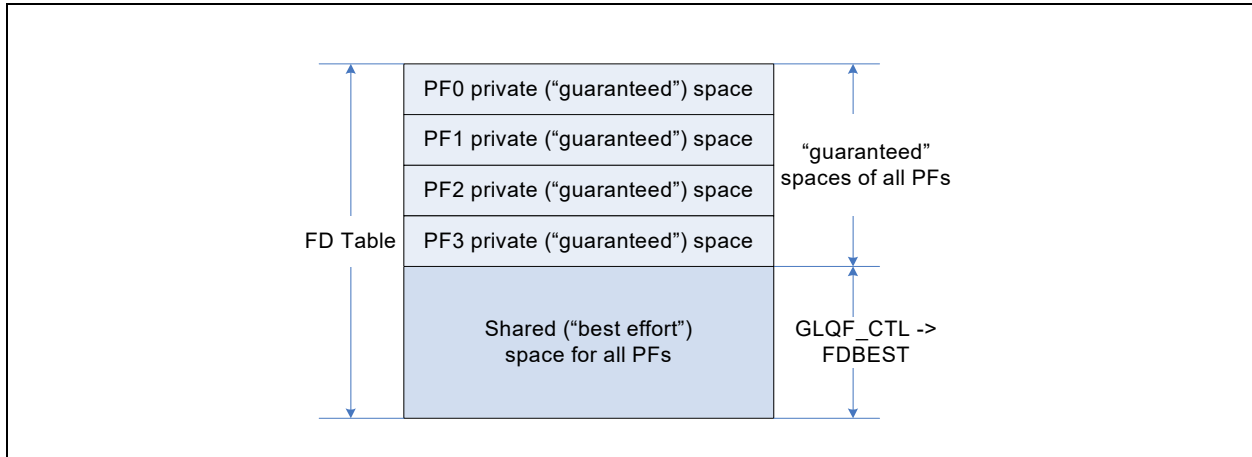


Figure 7-4. Flow Director Table Allocation (Example for 4 x PFs: PF0 ... PF3)

7.1.9.2 Statistic Counters

The FD filter might count packets by dedicated GLQF_PCNT counters. A filter can be associated to any of the GLQF_PCNT counters allocated to the PF by setting the *CNT_ENA* flag and the *CNT_INDEX* field in the Filter Programming Descriptor. The statistics counters are allocated statically to the PFs according to the *PCNT_ALLOC* setting in the GLQF_CTL register, as follows:

PCNT_ALLOC Setting	Enabled PFs	GLQF_PCNT Counters per PF	GLQF_PCNT Index Range per PF	Physical GLQF_PCNT Index Device Space
000b	Any Number	Flexible	0 ... 511	All 9 bits of the index provided at filter programming. Allocation to PFs should be handled by software. Possible ways to address it is by the NVM setting readable to software or any side band messages between the PFs that are outside the scope of this document.
100b	PF0 and PF1	256 = 512 / 2	0 ... 255	MS bit of the index is the LS bit of the PF index. 8 LS bits of the index equals the 8 LS bits of the filter index provided at filter programming.
101b	PF0 ... PF3	128 = 512 / 4	0 ... 127	2 MS bits of the index are the 2 LS bit of the PF index. 7 LS bits of the index equals the 7 LS bits of the filter index provided at filter programming.
110b	PF0 ... PF7	64 = 512 / 8	0 ... 63	3 MS bits of the index are the 3 LS bit of the PF index. 6 LS bits of the index equals the 6 LS bits of the filter index provided at filter programming.
111b	PF0 ... PF15	32 = 512 / 16	0 ... 31	4 MS bits of the index are the 4 LS bit of the PF index. 5 LS bits of the index equals the 5 LS bits of the filter index provided at filter programming.

7.1.10 Hash Functions

The X710-TM4 supports several hash functions to be used by the various filters:



Microsoft* Toeplitz-based hash vs. simple hash. These hash methods are described in the following sections. Selection between the two schemes is controlled by the global GLQF_CTL register as follows:

— HTOEP controls the scheme used by the hash filters for all PCTYPE's

- Symmetric hash (see [Section 7.1.10.3](#))

The following notation is used to describe the hash functions:

- Ordering is little endian in both bytes and bits. For example, the IP address 161.142.100.80 translates into 0xa18e6450.
- 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.

7.1.10.1 Microsoft Toeplitz-based Hash

This hash uses a random secret key of 416 bits (52 bytes). The key is defined per function by the PFQF_HKEY and VFQF_HKEY registers for the PFs and VFs, respectively. The algorithm operates by examining each bit of the hash input from left to right. Intel's nomenclature defines left and right for a byte-array as follows: given an array K with k bytes, nomenclature assumes that the array is laid out as follows: K[0] K[1] K[2] ... K[k-1].

While K[0] is the left-most byte, and the MSB of K[0] is the left-most bit. K[k-1] is the right-most byte, and the LSB of K[k-1] is the right-most bit.

```
ComputeHash(input[], N)
```

```
    For input[] of length N bytes (8N bits) and a random secret key K of 416 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; // cyclic shift while the right-most bit of K gets the left-  
most bit of K
```

```
    }
```

```
    return Result;
```

7.1.10.1.1 Pseudo-code Examples

The following four pseudo-code examples are intended to help clarify exactly how the hash is performed in four cases, IPv4 with and without ability to parse the L4 header, and IPv6 with and without a L4 header.



Table 7-11. Examples of Input Fields for the Hash function

Packet Type	Hash Input	Hash Result
IPv4 with TCP or UDP	SourceAddress, DestinationAddress, SourcePort, DestinationPort: Input[12] = @12-15, @16-19, @20-21, @22-23	Result = ComputeHash(Input, 12)
IPv4 without L4	SourceAddress, DestinationAddress: Input[12] = @12-15, @16-19	Result = ComputeHash(Input, 8)
IPv6 with TCP or UDP	SourceAddress, DestinationAddress, SourcePort, DestinationPort: Input[12] = @8-23, @24-39, @40-41, @42-43	Result = ComputeHash(Input, 36)
IPv6 without L4	SourceAddress, DestinationAddress: Input[12] = @8-23, @24-39	Result = ComputeHash(Input, 32)

7.1.10.1.2 RSS Verification Suite

This section provides a verification suite used to validate that the hash function is computed according to Microsoft nomenclature.

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,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00
```

Table 7-12. 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

The IPv6 address tuples are only for verification purposes, and might not make sense as a tuple.

Table 7-13. IPv6

Destination Address / Port	Source Address / Port	IPv6 Only	IPv6 with TCP
3ffe:2501:200:3::1 / 1766	3ffe:2501:200:1fff::7 / 2794	0x2cc18cd5	0x40207d3d
ff02::1 / 4739	3ffe:501:8::260:97ff:fe40:efab / 14230	0x0f0c461c	0xddde51bbf
fe80::200:f8ff:fe21:67cf / 38024	3ffe:1900:4545:3:200:f8ff:fe21:67cf / 44251	0x4b61e985	0x02d1feef



7.1.10.2 Simple Hash

Simple hash is an alternative hash function that trades runtime for effectiveness. It is provided for applications that recompute the hash in software and wish a lighter version of the hash function. Simple hash provides a more skewed distribution vs. the Microsoft hash. It is enabled only for the hash filter by the HTOEP flag in the global GLQF_CTL register.

Simple hash is performed by partitioning the sequence of input bits into 32-bit entities and XOR those to generate a 32-bit output.

```

ComputeSimpleHash(input[], N)
    For input[] of length N bytes (8N bits)
    Result = 0;
    For each DWord (32 bits) in input[] {
        Result ^= (left-most 32 bits of input);
        shift input left 32 bit position;
    }
return Result;

```

The input[] vector is padded by 0 to 3 bytes of zero's making it the entire number of DWords. In those cases that the hash function is used as an address to a lookup table smaller or equal then 64K entries, the 16 MS bits of the results are XORed with the 16 LS bits creating a 16 bit output. In case that the hash function is used as an address to a lookup table smaller or equal than 256 entries, the 4 bytes of the above results are XORed together creating an 8 bit output.

7.1.10.3 Symmetric Hash

A symmetric hash provides the same value if its respective source and destination fields are swapped. For example, suppose that the hash is done on the frame source and destination IP addresses. A symmetric hash guarantees that: $\text{hash}(\text{src IP}, \text{dst IP}) = \text{hash}(\text{dst IP}, \text{src IP})$.

The motivation behind symmetric hash is to route frame between two addresses to the same queue independent of the direction of transmission.

The symmetric hash is useful for field pairs like the IP addresses, L4 port numbers, MAC addresses, FC_IDs and FC exchange IDs. The symmetric hash is obtained by replacing the original source and destination fields by a XOR value of these fields before calculating the hash.

Symmetric hash is enabled per PCTYPE by the GLQF_HSYM and per port by the HSYM_ENA flag in the PRTQF_CTL_0 register.

7.1.10.3.1 Symmetric Hash Example for IPv4 with TCP

Non Symmetric input vector is: Input[12] = @12-15, @16-19, @20-21, @22-23.

Symmetric input vector is: Input[12] = @12-15 ^ @16-19, @12-15 ^ @16-19, @20-21 ^ @22-23, @20-21 ^ @22-23.



7.1.11 Rx Filter Admin Queue Commands

7.1.11.1 Add Tunneling UDP (0x0B00)

Add Tunneling UDP admin command (Table 7-14) is used to define a tunneling UDP filter. It defines the tunneling UDP port number and its header size, associating the filter to the PF that initiated the add command. The device response to this command is described as follows:

- If the requested UDP port is already defined for the PF, then report a completion with EEXIST error code.
- Else, if the requested UDP port is already defined with different Tunneling UDP Protocol Type then report a completion with an EMODE error code.
- Else, if the requested UDP port is already defined with matched parameters for other PF(s) then update the filter entry:
 - Set the matched PF flag of the filter
 - Report back a successful programming completion
- Else, if the requested UDP port does not exist then add a new filter entry:
 - Search for a free entry. If there is no free entry in the table then report completion with an ENOSPC error code.
 - Else, add the entry to the table:
 - Set the matched PF flag of the filter
 - Report back a successful programming completion

Table 7-14. Add Tunneling UDP Command (Opcode: 0x0B00)

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0B00	Command opcodes.
Datalen	4-5	0x00	N/A (reserved zero).
Return Value/ VFID	6-7	0x00	N/A (reserved zero).
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
UDP Port	16-17		A 16-bit value of the UDP port number to be added (while byte 17 is the MS byte which is first on the wire). Note that a zero value is a reserved value and should not be used by this command.
Reserved	18		Reserved.
Reserved	19		Reserved.
Reserved	20		Reserved.
Tunneling UDP Protocol Type	21		Tunneling UDP type should be programmed as follow: 0x00 = VXLAN key. 0x01 = Geneve key (next protocol is defined in the header). 0x02 = Reserved option for MAC in UDP tunneling. 0x10 = Teredo. 0x11 = VXLAN-GPE (next protocol is defined in the header). Else, reserved.
Reserved	22-31		Reserved.



Table 7-15. Completion for the Add Tunneling UDP Command (Opcode: 0x0B00)

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0800	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7		Some comments on specific errors (see Section 7.10.8 for the errors encoding): No Error = No error. EEXIST = Add filter rejected because it already exist. ENOSPC = Add filter rejected because of no space. EMODE = Add filter rejected because the filter parameters in the Add Tunneling UDP command do not match the programmed parameters in an existing filter with the same UDP port.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
UDP Port	16-17		A 16-bit value of the UDP port number that was added.
Filter Entry Index	18		Index of the added tunneling UDP filter (0 to 15). This index is assigned by the device to be used by the PF to remove the filter.
Multiple PFs	19		This field equals to 0x1 when other PF(s) own the same filter entry and equals to 0x0, otherwise.
Total Filters	20		Total number of tunneling UDP filters used by all PFs after the completion of the Add Filter command.
Reserved	21-31		Reserved.

7.1.11.2 Remove Tunneling UDP (0x0B01)

Remove Tunneling UDP admin command ([Table 7-17](#)) is used to delete a tunneling UDP filter from the table. The device response to this command is described as follows:

- If the requested filter (defined by the Filter Entry Index) is not owned by the PF then report an ENOENT error code.
- Else (the requested filter to be removed is owned by the PF), clear an internal flag that associates the filter entry with the PF and report back successful filter removal completion.
 - If the filter is not associated with any PFs then set the *Entry Free* flag in the reported completion status.

Table 7-16. Remove Tunneling UDP Port Command (Opcode: 0x0B01)

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0B01	Command opcodes.
Datalen	4-5	0x00	N/A (reserved zero).
Return Value/ VFID	6-7	0x00	N/A (reserved zero).
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.



Table 7-16. Remove Tunneling UDP Port Command (Opcode: 0x0B01)

Name	Bytes.Bits	Value	Remarks
Reserved	16-17		Reserved, zero.
Tunneling UDP Filter Index	18.0:3		Filter entry index to be removed (0 to 15).
Reserved	18.4-31		Reserved, zero.

Table 7-17. Completion for Remove Tunneling UDP Command (Opcode: 0x0B01)

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x801	Command opcode.
Datalen	4-5	0x00	N/A
Return Value/ VFID	6-7		Some comments on specific errors (see Section 7.10.8 for the errors encoding): No Error = No error. ENOENT = Remove filter rejected because no matched entry was found.
Cookie High	8-11	Cookie	Opaque Value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque Value. Copied by firmware into the completion of this command.
UDP Port	16-17		A 16-bit value of the UDP port number.
Filter Entry Index	18		The index of the requested filter entry to be removed (between 0 to 15).
Multiple PFs	19		This field equals to 0x1 when other PF(s) still own the same filter entry and equals to 0x0, otherwise.
Total Filters	20		Total number of tunneling UDP filters used by all PFs after the completion of the Remove Filter command.
Reserved	21-31		Reserved, zero.



7.2 L2 Packet Processing

7.2.1 CRC Handling

7.2.1.1 Ethernet CRC Insertion

The X710-TM4 calculates and inserts the Ethernet CRC for all packets transmitted to the network according to a per-port setting configured by setting the *CRC Enable* bit of the Set MAC Config Admin Queue command.

7.2.1.2 Ethernet CRC Stripping

The X710-TM4 checks the integrity of the Ethernet CRC and possibly strips it. On packets that are posted to LAN queues, The Ethernet CRC bytes are stripped according to setting of the CRCStrip flag in the target LAN queue context.

7.2.2 L2 Padding

Transmit packets to the network are padded with zero's to 60 bytes guaranteeing that their length including the CRC is at least 64 bytes (see [Section 3.2.1.3](#)).

Receive packets to the host are padded as well. Packets are padded with zero guaranteeing that they are never shorter than 60 bytes if the following conditions are met:

- Received packet to host memory from the network with no CRC bytes (stripped by the device) and additional stripped fields (like VLAN) that their remaining length is shorter than 60 bytes.
- Loopback packets to host memory (VM-to-VM) with or without stripped fields (like VLAN) that their remaining length is shorter than 60 bytes.
- Same rules for packets destined to the EMP.

7.2.3 L2 Tag Handling

7.2.3.1 Overview

In [Section 7.2.4.4](#), different options to manipulate the L2 tags are described. This section describes the generic mechanism used to handle L2 tags in the X710-TM4.

The X710-TM4 supports up to 8 tags. In general, the following text applies to each tag independently.

The order these tags are expected in a packet is according to their index in the array listed in [Table 7-20](#). Index 7 is the most inner L2 tag (closest to the L3 header) and index 0 is the most outer L2 tag (closest to the MAC address) as shown in [Figure 7-5](#).

Note: The following text refers to offloads provided by the device. A packet received from the network can have any number of tags and a packet transmitted might have any number of tags added by software provided no offload is required.

Note: In the sections that follow, when priority bits are mentioned it refers to the 3 MSBs of the tag. See [Section 7.2.3.4](#) for the programming interface to describe the supported tags and the way to handle them.

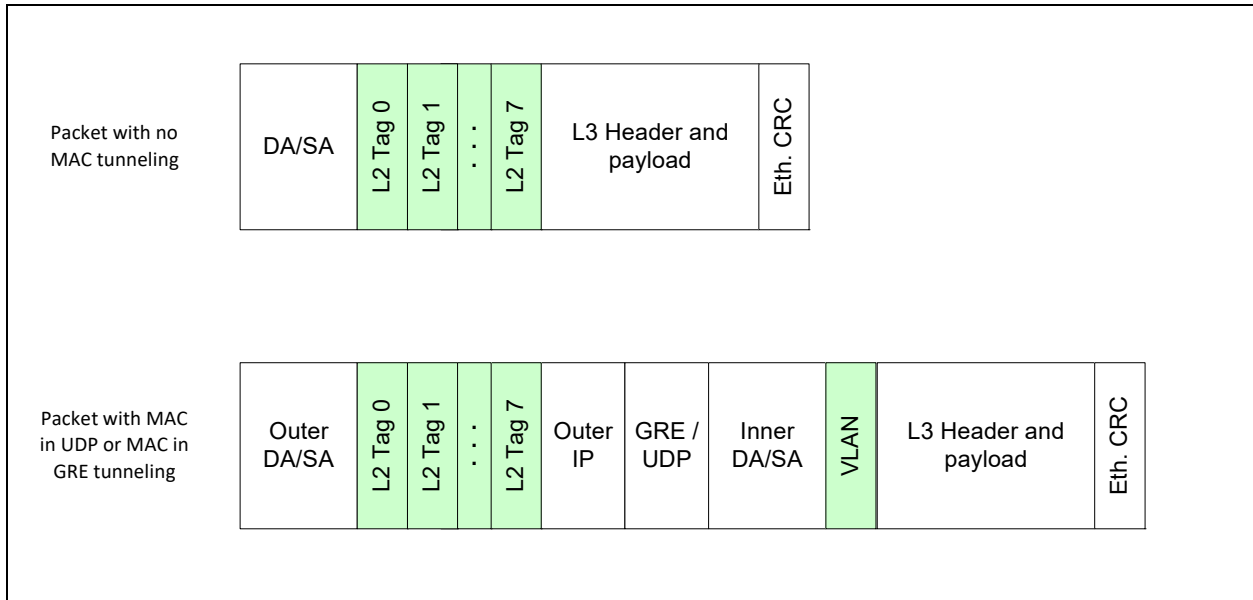


Figure 7-5. L2 Tags Order

7.2.3.2 Transmit Tag Handling

The transmit tag handling has three logical parts as shown in [Figure 7-6](#).

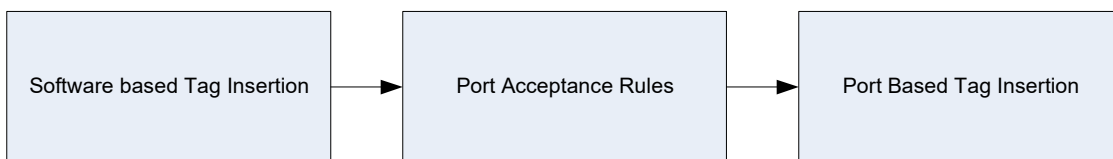


Figure 7-6. Transmit Tag Handling

The software-based tag insertion is described in [Section 7.2.3.2.1](#). The port acceptance rules and port based tag insertion are described in [Section 7.2.3.2.2](#).

[Section 7.2.3.3](#) describes the receive tag handling and [Section 7.2.3.4](#) describes the software interface used to manipulate the tags.



7.2.3.2.1 Software-based Tag Insertion Rules

Software can require to send packets with different L2 tags and might do so either directly through the data for all the tags or using the transmit descriptor as described in [Section 8.4.2.1](#) for up to two tags.

The tags available for the descriptor-based insertion are fixed by the device according to the mode of operation.

The type of tag taken from the *L2TAG1* field in this descriptor when the *IL2TAG1* field is set is defined by the *VSI_L2TAGSTXVALID.L2TAG1INSERTID* field when *VSI_L2TAGSTXVALID.L2TAG1INSERTID_VALID* is set. If the type of the L2 tag requires more than 2 variable bytes, additional bytes are taken from the *L2TAG2* field while the *L2TAG1* field is first on the wire. In this case, the *IL2TAG2* flag must be cleared.

The type of the tag taken from the *L2TAG2* field in this descriptor when the *IL2TAG2* field is set is defined by the *VSI_L2TAGSTXVALID.L2TAG2INSERTID* field when *VSI_L2TAGSTXVALID.L2TAG2INSERTID_VALID* is set.

After the tags are inserted according to the software request, the rules per VSI described in [Section 7.2.3.2.2](#) are applied to decide if the packet can be sent.

7.2.3.2.1.1 Single Tag Handling

Table 7-18 lists the tag insertion in different cases when each tag is identified by a different Ethertype.

Table 7-18. Single Tag Handling

Data Buffer Tag	Tx Descriptor Tag	Software Requested Action
No	No	Do nothing (send untagged packet)
No	Yes	Insert Tag from the descriptor
Yes	No	Do Nothing (send packet with the tag from the buffer)
Yes	Yes	This is not a valid configuration and should not be used.

Note: The previous table relates to the configuration of a single tag. The configuration of different tags is independent.

7.2.3.2.1.2 Double Tag Handling

If the L2 tags configuration table contains two tags with the same Ethertype (such as two VLANs with Ethertype=0x8100) and both tags are enabled, there might be some cases where hardware might not be able to identify which of the two tags were inserted by the software device driver directly in the packet.

In this case, if only one tag with that Ethertype is seen in the packet and no insertion was requested by the software device driver or via port-based insertion, it is assumed to be the first of the two tags (outer VLAN example). Any further action (anti spoof, UP translation, tag accept policy and port-based tagging) interprets this tag according to this. For example, if anti spoof is applied only to the inner VLAN, a packet with a single VLAN is treated as a packet with no VLAN for anti spoof.

If a single tag of these two is present in the buffer sent by the host, but the descriptor request insertion of one of the tags (like an outer VLAN), assume that the tag present in the buffer is the tag for which offload was not requested (in this case inner VLAN).



After the decision on the identity of each tag as previously described, the handling of each tag is as described in Section 7.2.3.2.1.1.

Note: If a tag is inserted by the software device driver or using port-based insertion, the previous explanation is not relevant and the tag is offloaded as required.

7.2.3.2.2 Port-based (VSI) Tag Handling

7.2.3.2.2.1 Accept Rules

The previous section described how tags are inserted via the data buffer or the transmit descriptor. This section describes the rules applied per-VSI to decide if a software request is acceptable. Table 7-19 lists the applied rule according to tag accept, tag insert and the tag presence in the packet for a specific tag.

Table 7-19. Port-based Tag Handling (Transmit)

Tag Accept Mode	Tag Inserted by Software Device Driver?	Port-based Tag	Allowed Software Device Driver Behavior ¹	Action
Allow untagged only: VSI_TAR.ACCEPTTAGGED = 0b ACCEPTUNTAGGED = 1b	No	No	Yes	Send packet as is.
	No	Yes	Yes	Tag inserted by the VSI.
	Yes (from descriptor and VLAN ID = 0)	Yes	No	VSI's VLAN tag overrides software while keeping the priority and DEI bits provided by software.
	Yes (in packet)	Yes	No	Drop packet ² .
	Yes	No	No	Drop packet.
	VLAN ID = 0 ³	No	Yes	Send packet as is.
	VLAN ID = 0 ³	Yes	Yes	Port-based (VSI's) VLAN tag overrides the tag provided by software while keeping the priority and DEI bits provided by software. This mode is supported only if the tag is inserted in the descriptor and only if the insertion is from the first descriptor tag and first VSI tag or from the second descriptor tag and second VSI tag.
Allow tagged and untagged: VSI_TAR.ACCEPTTAGGED = 1b ACCEPTUNTAGGED = 1b	X	Yes	Unexpected VSI Setting	Undefined.
	X	No	Yes	Send packet as is.
Allow tagged only: VSI_TAR.ACCEPTTAGGED = 1b ACCEPTUNTAGGED = 0b	X	Yes	Unexpected VSI Setting	Undefined.
	No	No	No	Drop packet.
	Yes	No	Yes	Send packet as is.
Accept nothing: VSI_TAR.ACCEPTTAGGED = 0b ACCEPTUNTAGGED = 0b	X	X	X	Non-legal configuration.

1. An unexpected VSI setting means that the combination of tag accept mode and port-based tag should not be requested when creating a VSI,
2. This behavior is not implemented in the X710-TM4. The port-based VLAN inserted before the host-based VLAN and the packet is sent.
3. If GL_SWT_L2TAGCTRL.ISVLAN is set. Can only be set in one tag (inner VLAN).



7.2.3.2.2.2 Port-based Tag Insertion Mechanism

After the packet sent by the software device driver was accepted, the X710-TM4 might add up to three tags based on the VSI that sent the packets. Two of the added tags can have a variable part of up to 16 bits and one tag can have a variable part of up to 32 bits. The tags that are allowed to be port-based are determined via the VSI_L2TAGSTXVALID.TIR[012]_INSERT and VSI_L2TAGSTXVALID.TIR[012]_INSERTID bits.

The tags for which port-based insertion is done are fixed by the device according to the mode of operation.

For tags up to 8 bytes (not including the Ethertype), the entire tag can be inserted by hardware. Either as a request of software via the descriptor as described in [Section 7.2.3.2.1](#) or as part of a port-based tag insertion.

Each tag might contain the following parts:

- The Ethertype
- A fixed part
- A variable part (up to 16 or 32 bits).

The Ethertype is taken from the GL_SWT_L2TAGCTRL.ETHERTYPE field.

The fixed part is taken from the GL_SWT_L2TAGDATA0 and GL_SWT_L2TAGDATA1 register. The GL_SWT_L2TAGCTRL.LENGTH field indicates the length of the Ethertype payload.

Note: The unused part of the fixed part and the word in which the variable part is inserted should be set to zero.

The variable part is extracted from the descriptor or from the VSI_TIR register. The GL_SWT_L2TAGTXIB.OFFSET and the GL_SWT_L2TAGTXIB.LENGTH defines the location and length of the variable part. [Figure 7-7](#) shows the relationship between the different parameters.

The order of the tags inserted from the port-based tags is according to the order in the bit map. The first bit set uses the value from the VSI_TIR[0] register, the second bit set uses the value from the VSI_TIR[1] register and the third bit set uses the value from the VSI_TIR[2] register.

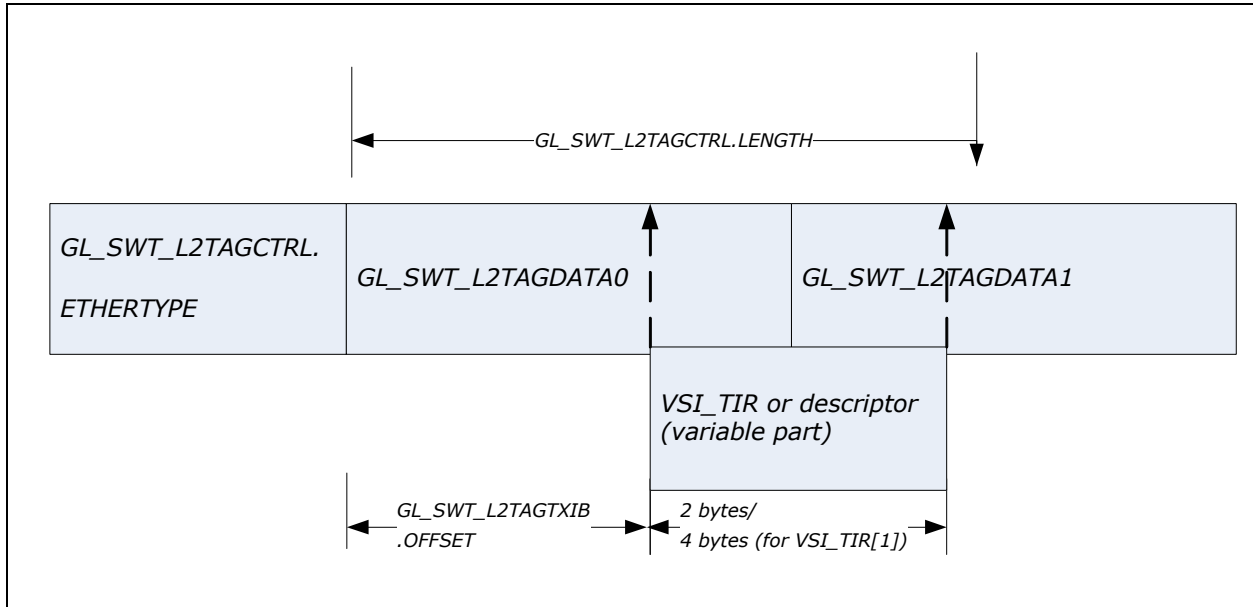


Figure 7-7. Tag Insertion

7.2.3.2.2.3 Queue-based Tag Insertion

For one of the tags, the X710-TM4 can insert different port-based tags based on the queue from which the packet was sent. If the ALT_VLAN bit in the transmit queue context is set, the tag is taken from the alternate tag register (VSI_TAIR instead of being taken from VSI_TIR).

This capability is available for each VSI in the tag inserted to the VSI_L2TAGSTXVALID.TIROINSERTID tag.

7.2.3.3 Received Tag Extraction Rules

The tag extraction from receive packets and insertion in the receive descriptor write back is controlled by the VSI_TSR.STRIPTAG, VSI_TSR.SHOWTAG, and VSI_TSR.SHOWPRIONLY bit fields. The options supported for each tag are:

- Do nothing.
- Remove this tag from receive packets and do not insert into descriptor.
- Remove this tag from receive packets and insert into descriptor.
- Remove this tag from receive packets and insert only priority bits into descriptor.

Note: The DEI bit is considered part of the VLAN tag and is hidden if the VLAN tag is hidden.

Up to two L2 tags can be extracted into the Rx descriptor. One of the tags can supply up to 32 bits of data and the other tag can supply up to 16 bits of data (a total of three 16-bit words maximum). If more than one word needs to be extracted, 32-byte descriptors need to be used (the Dsize flag in the receive queue context is set to 32B_Descriptor); otherwise, 16-byte descriptors can be used (the Dsize flag is set to 16B_Descriptor).



The order of the tags extracted to the descriptor is according to the order in the VSI_TSR.SHOWTAG bitmap and according to the L2TSEL field in the queue context. If the L2TSEL bit is set, the first bit set extracts the value to the L2TAG1 field in the receive descriptor. The second bit set extracts the value to the L2TAG2 (2nd) and L2TAG2 (1st) fields in the receive descriptor. If the L2TSEL bit is cleared, the first bit set extracts the value to the L2TAG2 (2nd) and L2TAG2 (1st) field in the receive descriptor. The second bit set extracts the value to the L2TAG1 field in the receive descriptor.

Note: The L2TAG2 (1st) field is used only if the GL_SWT_L2TAGRXEB.LENGTH is 10b or 11b (24 or 32 bits extracted part); otherwise, the entire tag is stored in L2TAG2 (2nd).

When removing the packet of a tag is requested, the total L2 tag (determined by the GL_SWT_L2TAGCTRL [7:0].LENGTH field), including Ethertype, is extracted from the packet.

Further detail on the reporting in the descriptor can be found in [Section 8.3.2.2](#).

Warning: Extracting two tags should be used only if the software device driver knows which tags are expected in the packets.

7.2.3.4 Tag Handling (Programming Interface)

This section describes the CSR interface available to control the L2 tags handling of the X710-TM4. The actual programming of these capabilities is either through an NVM image loading of the device-wide behavior, through port-wide commands such as the Set Port Parameters command for the port-wide behavior or through the Add VSI admin command ([Section 7.4.9.5.5.1](#)) for the per-VSI behavior.

The X710-TM4 support up to 8 types of programmable L2 tags. The types of L2 tags are defined in the NVM and in registers. For each tag the following parameters are defined:

Table 7-20. L2 Tag Control Registers (Global)

Register	Field	Description	Register Description
GL_SWT_L2TAGCTRL [7:0]	ETHERTYPE	The Ethertype of the L2 tag.	Section 10.2.1.7.1
	ISVLAN	This tag a VLAN tag. This information is used to define if priority tagging should be supported for this tag.	
	INNERUP	If this bit is set, the UP remapping is done on this field. If this bit is set, ISVLAN should also be set. If set in multiple tags, then the inner UP is taken from the first tag with this bit set in the packet.	
	OUTERUP	If this bit is set, this is the tag on which the inner to outer UP remapping is applied. Should be set only in one tag.	
	LENGTH	The length of the L2 tag (not including the Ethertype). The length can be 2, 4, 6 or 8 bytes.	
	HAS_UP	Defines if this tag includes UP bits that should be used for UP to TC translation. The first such tag found in the packet is used for UP to TC translation.	
GL_SWT_L2TAGTXIB [7:0]	OFFSET	Describes the offset in the header to which the variable data should be inserted (can be up to 8 bytes).	Section 10.2.1.2.12
GL_SWT_L2TAGDATA0[7:0], GL_SWT_L2TAGDATA1[7:0],	L2TAGDATA	The fixed part to insert in Transmit packets.	Section 10.2.1.2.14 and Section 10.2.1.2.15

Each port defines which tags to expect in packets sent and received through this port using the PRT_L2TAGSEN.ENABLE field.



The GLGEN_MAC_LINK_TOPO register from each VSI defines which of the 8 L2 tags are offloaded for its Tx and Rx traffic using the VSI_L2TAGSTXVALID and VSI_TSR registers. These registers define which receive tags and transmit tags to handle.

Each VSI might be configured with a different behavior for each of the tags. The possible behaviors relates to the type of tags the software device driver is allowed to insert in transmit packets, the type of tags inserted by the switch, the tags removed from received packets and the tags posted to the receive descriptor write back.

Table 7-21 lists the register fields used to set the expected behavior for each tag.

Table 7-21. L2 Tag Control Registers (Per VSI)

Register	Field	Description	Register Description
VSI_L2TAGSTXVALID	L2TAG[12]INSERTID ¹ , L2TAG[12]INSERTID_VALID	Defines the tags for which descriptor-based insertion is supported. The ID is based on the L2 tags mapping listed in Table 7-22.	Section 10.2.1.10.12
	TIR[012]INSERTID, TIR[012]_INSERT	Defines the tags for which port-based insertion is supported.	
VSI_TAR ²	ACCEPTTAGGED[n]	A bitmap describing if a packet with tag 'n' is accepted.	Section 10.2.1.9.1
	ACCEPTUNTAGGED[n]	A bitmap describing if a packet without tag 'n' is accepted. ^{3,4}	
VSI_TIR_n (n = 0..2)	PORT_TAG_ID	The tag to insert.	Section 10.2.1.10.7
VSI_TAIR	PORT_TAG_ID	The alternate tag that can be used instead of VSI_TIR[0].	Section 10.2.1.10.7
VSI_TSR	STRIPTAG[n] (n = 0..9)	Defines if the tag should be extracted from the packet.	Section 10.2.1.10.11
	SHOWTAG[n] (n = 0..9)	Defines which of the tags should be extracted to the descriptor. Valid only if the corresponding bit in <i>STRIPTAG</i> is set. The <i>SHOWPRIONLY</i> field defines which part of the tag to extract to the descriptor. At most, two of these bits should be set. If more than two bits are set, only the first two are considered.	
	SHOWPRIONLY[n] (n = 0..9)	A per-tag bitmap defining which part of the tags to extract to the descriptor. If set, only the priority bits are extracted; otherwise, the entire tag is used. Relevant only if the corresponding bit in <i>SHOWTAG</i> is set.	

1. In order to insert priority bits from a descriptor and the VLAN tag value from hardware as listed in Table 7-18, the same ID should be used for L2TAG1INSERTID and TIR0INSERTID or L2TAG2INSERTID and TIR1INSERTID.
2. The tag number in this register relates to the 10 tags defined in Section 7.2.3.5.
3. If GL_SWT_L2TAGCTRL.ISVLAN is set, admits also priority tagged packets (VLAN tag = 0).
4. This bit should be set for all the tags not expected in the packet.

7.2.3.5 L2 Tags Configuration

This section describes the value to set in the different registers to implement the X710-TM4 POR.

The following L2 tags are currently defined:



Table 7-22. L2 Tags

Tag	Tag ID (in Table)	Tag ID in VSI_TAR and VSI_TSR Bitmaps
Reserved	0	2
S-tag	1	3
Outer VLAN	2	4
VLAN	3	5
Reserved	Reserved	Reserved

Table 7-23 lists the configuration for each of the tags:

Table 7-23. L2 Headers Support

Register/Tag	Reserved	S-tag	Outer VLAN	VLAN	Reserved	Reserved
Index	0	1	2	3	4	5-7
Global Configuration						
GL_SWT_L2TAGCTRL. ETHERTYPE		0x88A8 ^{1,2}	0x8100 ²	0x8100 ²		0
GL_SWT_L2TAGCTRL. ISVLAN		0	0	1		0
GL_SWT_L2TAGCTRL. INNERUP		0	0	1		0
GL_SWT_L2TAGCTRL. OUTERUP		1	0	0		0
GL_SWT_L2TAGCTRL. LONG		0	0	0		0
GL_SWT_L2TAGCTRL. HAS_UP		1	1	1	0	0
GL_SWT_L2TAGCTRL. LENGTH		2	2	2	6	0
GL_SWT_L2TAGTXIB. OFFSET		0	0	0		0
GL_SWT_L2TAGTXIB. LENGTH ³		01b	01b	01b		0
GL_SWT_L2TAGRXEB. OFFSET		0	0	0		0
GL_SWT_L2TAGRXEB. LENGTH ³		01b	01b	01b		0
GL_SWT_L2TAGDATA0, GL_SWT_L2TAGDATA1	All zeros.					
Per Port Configuration Controlled by Firmware						
PRT_L2TAGSEN.ENABLE		0/1	0/1 ¹	1	0	0
Per VSI Configuration Controlled by Add VSI Command						
VSI_TSR. STRIPTAG		0/1 ⁴	0	0/1 ⁵		0

**Table 7-23. L2 Headers Support**

Register/Tag	Reserved	S-tag	Outer VLAN	VLAN	Reserved	Reserved
VSI_TSR.SHOWTAG		0/1	0	0/1 ⁶		0
VSI_TSR.SHOWPRIONLY		0	0	0/1 ⁷		
VSI_TAR.ACCEPTTAGGED		0/1	1	0/1 ⁸		
VSI_TAR.ACCEPTUNTAGGED		1	1	0/1 ⁹		

1. If the *Channel Identifier* field in the features enable word in the EMP SR settings module header is set, the value of tag 1 is 0x8100 and tag 2 (outer VLAN) is not available.
2. This value can be changed using the [Set Switch Configuration Command](#).
3. 00b = 8 bits, 01b = 16bits, 10b = 24 bits, 11b = 32bits.
4. Should be set.
5. Should be set if VSI is not VLAN aware or VM requested VLAN extraction offload.
6. Should be set if VSI is VLAN aware and VM requested VLAN extraction offload.
7. Should be set if VSI in non VLAN aware but is DCB aware.
8. Should be set for VLAN aware VSIs.
9. Should be set if VSI must add VLAN.

Note: PRT_L2TAGSEN.ENABLE should not enable outer VLAN and S-tag concurrently.

7.2.4 VLAN Handling

This section describes the handling of IEEE 802.1Q VLAN tags based on the features previously described. Handling of other L2-tags as S-tag are described in the switching chapter ([Section 7.4](#)).

There can be up to two VLAN tags in the outer header of a packet identified as VLAN (tag index 3) and outer VLAN (tag index 2).

In addition, the tunneled header might also include a VLAN. The handling of the tunneled VLAN is described in [Section 7.2.4.3](#).

Each port can be set to expect packets with outer VLAN using the Set Port Parameters admin command ([Section 7.4.9.5.3.4](#)). When enabled, a packet with a single VLAN is treated only as a packet with outer VLAN; otherwise, a single VLAN in a packet is treated as inner VLAN.

In any case, the outer VLAN is not part of the forwarding decision and should be handled by the software device driver or by the MC both in transmit and receive. There is no offload of outer VLAN insertion or extraction.

The following sections describe the handling of the inner VLAN.

In UDP tunnels, an internal VLAN might also be present as shown in [Figure 7-26](#). The handling of this VLAN is described in [Section 7.2.4.3](#).

UP translation in inner VLAN is described in [Section 7.2.6.2](#).

7.2.4.1 Transmit Flow

This section describes the handling of VLAN as part of the flow of packets sent by the host, MC or EMP.



7.2.4.1.1 Tag Insertion

A VLAN tag can be inserted to the packets in three ways:

1. As part of the packet buffer.
2. As part of the transmit descriptor in the *L2TAG1* field if the *IL2TAG1* field is set.
3. By the device from the VSI context.

The first two options are enabled if the VSI is allowed to add a VLAN tag by the VLAN driver insertion mode in the Add VSI command ([Section 7.4.9.5.5.1](#)). If a packet is sent with a VLAN tag from a VSI that is not allowed to add a tag, it is dropped.

Note: If the *IL2TAG_IL2H* field in the transmit context descriptor is set, *L2TAG1* represents the inner VLAN and regular VLAN insertion from the descriptor is not available. See [Section 7.2.4.3.1](#) for details on the inner tag insertion. Inner tag insertion is allowed irrespective of the VLAN configuration in the Add VSI command.

The third option is enabled by setting *Insert PVID* in the same command. The tag to insert is defined in the *PVID+Default UP* field of the command.

Note: Setting both *Insert PVID* and *VLAN driver insertion mode* to enable software to insert VLAN is not allowed.

Note: It is expected that MC traffic arrives with the right VLAN tagging.

7.2.4.1.2 VLAN Anti Spoofing

After the packet is VLAN tagged by one of the previous methods, if the *Enable VLAN anti spoof* bit in the Add VSI command is set, it is compared to the ingress VLAN list as described in [Section 7.4.6.1.1.2](#) and dropped if the inserted VLAN is not in the list.

7.2.4.1.3 VLAN Filtering

If the packet passed the previous stage, the VLAN tag is used as part of the forwarding process. It is compared as part of the MAC, VLAN filters added by the Add MAC, VLAN pair command ([Section 7.4.9.5.9.1](#)) to determine if the packet should be sent to a local address or should be sent to the network. It is also compared to the PRT_MNG_MAVTV filters as part of the manageability filtering as described in [Section 9.3](#) to define if it should be sent to the MC.

7.2.4.2 Receive Flow

This section describes the handling of VLAN as part of the flow of packets received by the host, MC or EMP.



7.2.4.2.1 VLAN Filtering

When a packet is received from the network (or from the host) the VLAN tag is used as part of the forwarding process. It is compared as part of the MAC, VLAN filters added by the Add MAC, VLAN pair command (Section 7.4.9.5.9.1) and compared to the VLAN egress filtering set by the Add VLAN admin command (Section 7.4.9.5.9.3) to determine if the packet should be sent to a local VSI. It is also compared to the PRT_MNG_MAVTV filters as part of the manageability filtering as described in Section 9.3 to define if it should be sent to the MC.

7.2.4.2.2 VLAN Extraction

Before a packet is stored in host memory, the VLAN tag might be stripped and optionally stored in the receive descriptor. The action done is defined per VSI in the VLAN and *UP expose mode* (Rx) field in the Add VSI command. The possible actions are:

- Show VLAN and UP in descriptor (legacy behavior)
- Hide VLAN show UP in descriptor (VLAN ID exposed as zero)
- Hide VLAN and UP
- Do nothing (leave VLAN in packet)

If the VLAN or the UP are exposed in the descriptor, it shows in the *L2TAG1* field and the *L2TAG1P* flag is set. If the *L2TSEL* bit is set, the VLAN tag is extracted to the *L2TAG2* (1st) field in the receive descriptor instead. In this case, 32-byte descriptors must be used (*DSize* field in the receive queue context must be set).

Note: When a packet is sent to the MC, the VLAN is kept in the packet.

7.2.4.3 VLAN in Tunnel Packets

In MAC in UDP and MAC in GRE encapsulations, a VLAN within the tunneled MAC header might also be present as shown in Figure 7-26.

7.2.4.3.1 Inserting Tunneled VLAN From Descriptor

Inserting this tag is controlled via the *IL2TAG_IL2H* field in the transmit descriptor. When set, the *L2TAG2* field in the context descriptor contains the tunneled VLAN. In this case, the values of *VSI_L2TAGSTXVALID.L2TAG2INSERTID* and *VSI_L2TAGSTXVALID.L2TAG2INSERTID_VALID* fields are ignored.

This mode is enabled irrespective of the VLAN configuration in the Add VSI command.

7.2.4.3.2 Extracting Tunneled VLAN to Descriptor

The *SHOWIV* field in the receive queue context controls extracting an internal VLAN to the receive descriptor. If set, the tunneled VLAN is inserted in *L2TAG2* (1st) field of the receive descriptor write-back. If the *L2TSEL* bit is cleared, the inner VLAN tag is extracted to the *L2TAG1* field in the receive descriptor instead. When the *SHOWIV* field is set and the *L2TSEL* bit is set, 32-byte descriptors must be used (*DSize* field in the receive queue context must be set).



7.2.4.3 Tunneled VLAN in Pass Through Traffic

Tunneled VLAN is not offloaded (not inserted nor extracted) for MC pass through traffic.

7.2.4.4 Port-based VLAN

The X710-TM4 supports the port-based VLAN feature by enabling any VSI to specify the default VLAN that it belongs to. The port-based VLAN association is done when a packet received on the VSI is untagged or priority tagged and protocol VLAN association is not done. Port-based VLANs map packets received on a given VSI with the corresponding port-based VLAN identifier called PVID. The PVID list should be programmed with the port VLAN IDs for each VSI.

The port VLAN list is provided and is part of the VSI context. [Table 7-24](#) lists the port VLAN parameters in the port VLAN list and the matching parameters in the Add VSI command.

Table 7-24. Port VLAN List and VSI Parameters

Port Based VLAN list	Add VSI parameter	Notes
VSI number	VSI number	Returned by firmware in response.
PVID	PVID + default UP	
Default UP		
Admit 1Q tagged only	VLAN driver insertion mode	
Admit untagged/priority tagged only		
Admit all		
Ingress VLAN check enable	Enable VLAN anti spoof	
Insert PVID	Insert PVID	
Expose VID and UP of received packets	VLAN and UP expose mode (Rx)	
Expose UP only of received packets		
Do not expose VID or UP of received packets		
Ingress UP translation table	Ingress UP translation table	See Section 7.2.6.1 .
Egress UP translation table	Egress UP translation table	See Section 7.2.6.2 .

The switch should insert the PVID to the packet if the Insert PVID parameter is set for the VSI and replace the UP bits according to the algorithm described in [Section 7.2.6.1](#).

Once the VLAN ID association is made, the packet forwarding is done using the VLAN, MAC forwarding table and VLAN membership table.



7.2.5 Outer Tag Handling

This section describes the handling of outer VLAN tags (IEEE 802.1Qbg S-tags or 0x8100 VLAN tags) based on the features previously described.

There can be up to one outer tag in a packet (tag index 1). The sections that follow describe the handling of S-tag.

Note: The outer tag Ethertype can be either 0x88A8 or 0x8100 according to the *Channel Identifier* field in the features enable word in the EMP SR settings module header (0x0 = 0x88A8 and 0x1 = 0x8100).

7.2.5.1 Transmit Flow

This section describes the handling of outer tag as part of the flow of packets sent by the host, MC or EMP.

There is no anti spoofing capability for outer tags.

7.2.5.1.1 Tag Insertion

An outer tag can be inserted to the packets in three ways:

1. As part of the packet buffer.
2. As part of the transmit descriptor in the *L2TAG2* field if the *IL2TAG2* field is set.
3. By the device from the VSI context.

The first two options are enabled if the VSI is allowed to add an outer tag by the clearing the *S-tag insert enable* field and setting the *Accept tag from host* field in the Add VSI command (Section 7.4.9.5.5.1). If a packet is sent with an outer tag from a VSI not allowed to add a tag, it is dropped.

The third option is enabled by setting the *S-tag insert enable* in the same command. The tag to insert is defined in the *S-tag* field of the command.

Note: Setting both *S-tag insert enable* and *Accept tag from host* to enable software to insert VLAN is not allowed.

Note: As an outer tag of 0x8100 can be confused with an inner tag of the same value, it should be enabled in the Set Port Parameters command only if expected on all packets in this port.

7.2.5.2 Receive Flow

This section describes the handling of S-tag as part of the flow of packets received by the host, MC or EMP.



7.2.5.2.1 S-tag Extraction

Before a packet is stored in host memory, S-tag might be stripped and optionally stored in the receive descriptor. The action done is defined per VSI in the *S-tag extract mode* field in the Add VSI command. The possible actions are:

- Show S-tag and UP in descriptor (legacy behavior). Used for cascaded port virtualizer with offload.
- Hide S-tag and UP.
- Do nothing (leave S-tag in packet). Used for cascaded port virtualizer without offload.

If the S-tag is exposed in the descriptor then it shows in the *L2TAG2* field and the *L2TAG2P* flag is set. In this case, 32-byte descriptors must be used (*DSize* field in the receive queue context must be set).

Note: If the *SHOWIV* field is set in the queue context, the *L2TAG2* field is allocated for the inner VLAN of tunnel packet. In this case, S-tag is never inserted in the descriptor.

7.2.6 User Priority Bits (802.1p) Handling

The UP bits are used to differentiate between multiple classes of traffic. The X710-TM4 supports multiple use cases related to the handling of the UP bits:

1. An operating system that is not DCB/traffic type aware and uses a single TCID.
2. An operating system that is traffic type aware but not DCB aware. It distributes the traffic to different queues but cannot tag them with the right UP. This case refers to LAN flows. The operating system is not aware but the software device driver knows the difference and can assign the different types of traffic to different flows. There is no DCBX agent so the software device driver does not know the UP or TC for flows.
3. An un-trusted operating system that might set UP bits of TCs it is not allowed to use.
4. An operating system that uses a single queue per TCID but needs to use more UPs in order to gain from the differentiated QoS in the network.

The following sections describe the handling of UP bits in transmit and receive to support the previous use cases.

7.2.6.1 Transmit Functionality

Each transmit packet is assigned a UP, either by the software device driver or as part of the port VLAN table as described in [Section 7.2.4.4](#).

This UP is translated using two different translation vectors.

The first vector is specific to a VSI and is part of the port VLAN table (Transmit UP translation table - *VSI_TUPR*). This vector reflects the mapping of the user priorities as seen by the operating system to the user priorities as seen by the network.

The second vector is specific to a TCID in a physical port. It translates the UP received from the previous translation to a UP matching the TCID to which the packet is associated. These vectors can be configured through the *PRT_TCTUPR* registers.

Note: There is no drop of packets due to a wrong 802.1p tagging. If a wrong tag is requested, the tag is replaced. As a result, this functionality supports both UP anti-spoofing and port-based UP.

Untagged packets are kept as is. No translation is done on them.

Figure 7-8 shows the algorithm.

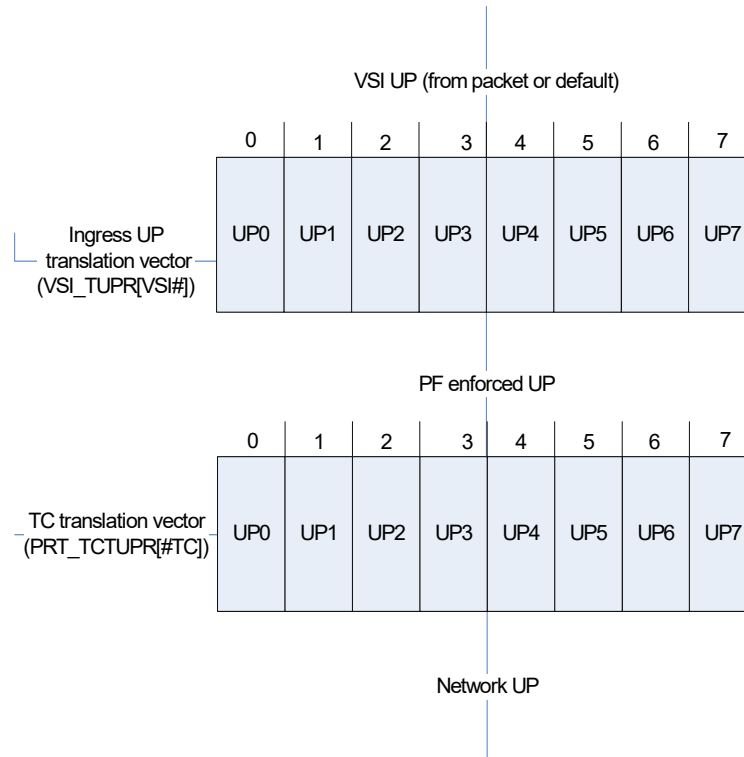


Figure 7-8. Non-VLAN aware/Non-DCB Aware Operating System Example

Assume a VF that queues LAN traffic and iSCSI traffic to different queues, but is not VLAN aware. In this case, the packet is sent by the software device driver with no VLAN and the default UP set in the port-based VLAN (like zero) is the initial UP of the packet. In this case, the ingress UP translation vector (VSI_TUPR) is not relevant and might be for example {0,1,2,3,4,5,6,7} or {0,0,0,0,0,0,0,0}. The TC translation vector (PRT_TCTUPR) for LAN might be {0,1,2,0,0,0,0,0} and the TC table for storage might be {4,4,4,4,5,4,4,4}. So a LAN packet goes out with a UP of 0 and a storage packet goes out with a UP of 4.

Example: Non DCBx Tagging Operation System

Assume a VF that tags LAN high-priority traffic with a UP of 4 and LAN low priority traffic with a UP of 3 and the storage traffic as UP 7. In this case, the VF translation table might be {0,1,2,0,2,4,4} and the TC tables as previously described. The low-priority LAN goes out with a UP of 0 and the high-priority LAN with a UP of 2 and storage with a UP of 4.

The tag on which the transmit UP translation is done is identified by setting the *INNERUP* bit in the matching GL_SWT_L2TAGCTRL register.



The default mapping of both tables is identity mapping (such as mapping a UP to itself). If UP translation is not needed these mapping should not be changed.

7.2.6.1.1 Transmit Outer Tag User Priority

The transmit outer tag user priority can be handled in three different ways:

1. If the outer tag is received in the packet or the descriptor (like cascaded S-comp), the UP should be part of the packet sent by the software device driver or inserted from the descriptor.
2. If the outer tag is inserted by hardware, the UP might be either:
 - a. The UP defined as part of the inserted tag in the VSI_TIR register.
 - b. A translation of the regular VLAN UP as defined in the VSI_TUPIOM register.

If the outer tag is defined as *OuterUP* in the GL_SWT_L2TAGCTRL register, the UP is based on the first tag for which the *InnerUP* field is set in the GL_SWT_L2TAGCTRL register (usually the VLAN tag; option 2.b); otherwise, it is based on the default (option 2.a).

If there is more than one tag in the packet with *OuterUP* set, the inner-to-outer UP translation should be done to the most outer tag with the *OuterUP* bit set.

The tag to which translation is applied can be either S-tag or external VLAN.

Figure 7-9 shows the translation process.

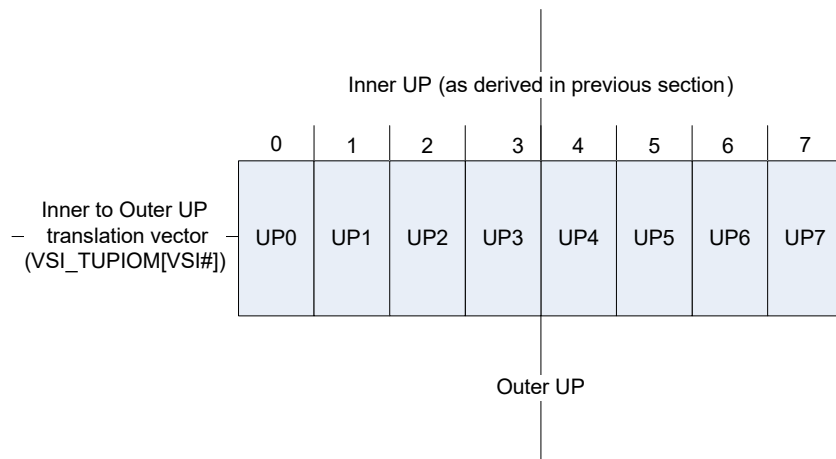


Figure 7-9. Tx Inner-to-Outer UP Translation



7.2.6.2 Receive Functionality

7.2.6.2.1 VLAN UP Translation

The priority bits of an inner VLAN tag in received packets might be remapped using the ingress UP translation table in the Add VSI command. This means the operating system might get packets with 802.1p priority bits reflecting the local configuration and not the link configuration. By default, the ingress UP translation section is not valid and the translation is one to one, meaning that the received UP is not translated.

7.2.6.2.2 VLAN UP Exposure to Software Device Driver

The exposure of the received UP to operating systems is defined by the awareness of the operating system controlling the VSI as follows:

1. For a monolithic operating system, VMM or for a guest operating system which is VLAN aware, the UP is exposed as part of the VLAN as requested by software device driver (either in the packet or in the descriptor).
2. If the guest operating system is DCB aware but not VLAN aware, it gets the priority bits as part of the receive descriptor in the VLAN tag field. The VLAN ID is zero, but the priority bits are valid.
3. A guest operating system that is not VLAN aware and not DCB aware does not get the user priority bits (UP) at all.

The configuration of the UP removal is part of the port VLAN configuration described in [Section 7.2.4.4](#) and in the Add VSI command VLAN handling section ([Section 7.4.9.5.5.1](#)).



NOTE: *This page intentionally left blank.*

7.3 Frame Formats

This section describes the packet formats supported by the device. It is structured by layers (L2, L3, and L4), providing the details per each layer. Packets that do not conform with the described formats are handled as L2 packets.

- The following restrictions apply on the lengths of parsed packets: The device only parses headers in the first 480 bytes of the packet. If the header (as defined in this section) extends beyond 480 bytes, the packet is handled as an L2 packet with a packet type of abort (0xFF).
- Each single header is limited to 255 bytes (unless restricted further in the text). If the header (as defined in this section) extends beyond 255 bytes, the packet is handled as an L2 packet with a packet type of abort (0xFF).
- Aborted packets by the parser are handled as L2 packets by the switch; a VSI decision is made by the outer MAC.

7.3.1 L2 Packet Formats

IEEE Std. 802.3-2018 and 802.3 SNAP packets are considered and treated as L2 packets. The X710-TM4 does not identify such packets explicitly in any way.

7.3.1.1 Standard L2 Packet Format

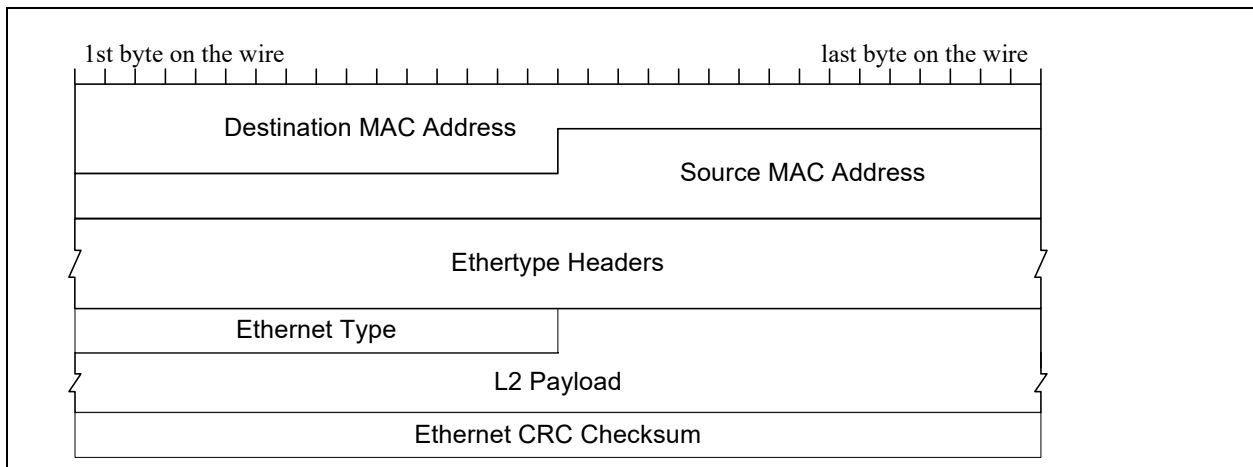


Figure 7-10. Standard L2 Frame Format

7.3.1.2 Serial Ethertypes (L2-tags)

Table 7-25 lists the supported Ether type headers. The order of Ether type headers in the L2 header is shown in Figure 7-11.

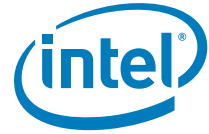


Table 7-25. List of Serial Ethertype Headers

Ethertype Header	Value	Length ¹	Where Defined
802.1Q S-tag	0x88A8	2 words	IEEE 802.1Q clause 9
Outer VLAN	Loaded from NVM. For example, 0x8100, 0x9100 and 0x9200.	2 words	IEEE 802.1Q clause 9
VLAN	0x8100	2 words	IEEE 802.1Q clause 9

1. Ethertype headers are an integer number of words. Length includes the *Type* field.

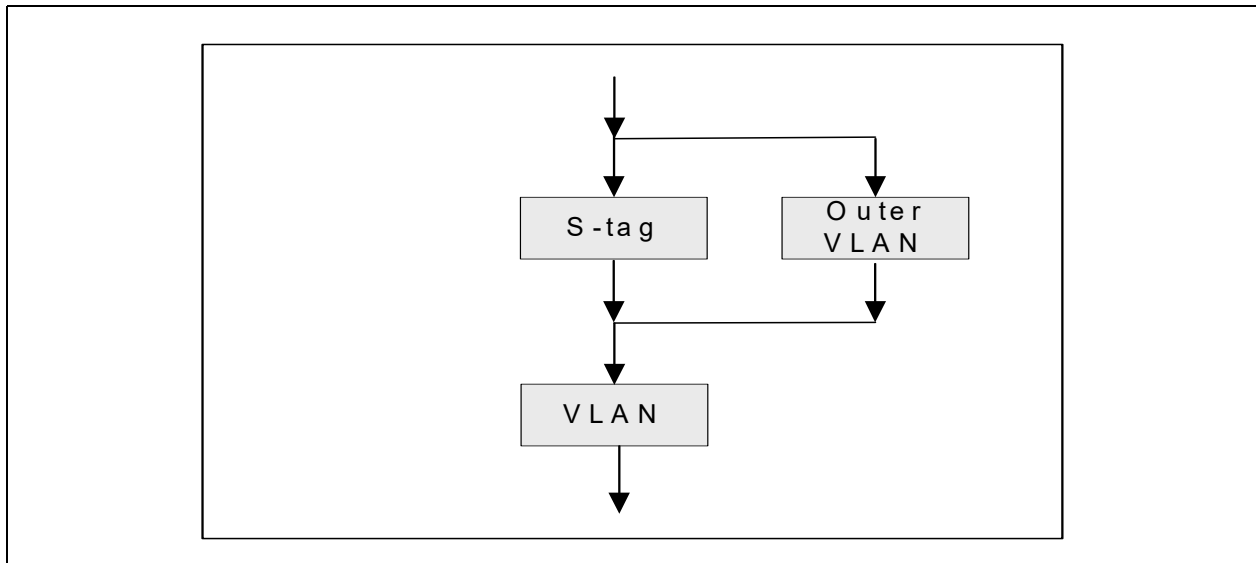


Figure 7-11. Order of L2 Ethertype Headers

7.3.1.3 Last Ethertype - Upper Layer Protocol (ULP)

The ULP following the L2 header is identified by the *Ethernet Type* field (see [Figure 7-10](#)).

[Table 7-26](#) lists the Ethertypes supported by the device along with their index value. A value of 00-00 means the entry is reserved.

Table 7-26. List of Ethernet Types

Index	Value	Purpose	Format of ULP
0	88-F7	IEEE 1588 (IEEE 802.1AS)	See Section 8.5 .
1	89-14	Reserved	
2	00-00	Reserved	
3	00-00	Reserved	
4	88-CC	IEEE 802.1ab (LLDP)	IEEE P802.1AB specification.

**Table 7-26. List of Ethernet Types**

Index	Value	Purpose	Format of ULP
5	88-47	MPLS Header	
6	00-00	Reserved	
7	89-4F	NSH Header	
8	88-8E	IEEE 802.1X (NAC)	N/A
9	08-06	ARP	See Section 7.3.2.6 .
10	00-00	Reserved	
11	00-00	Reserved	
12	08-00	IPv4	See Section 7.3.2.1 .
13	86-DD	IPv6	See Section 7.3.2.2 .
14	89-06	Reserved	
15	00-00	Reserved	

7.3.1.4 MPLS Header(s)

The X710-TM4 identifies MPLS headers. It skips these header's processing of the rest of the packet and providing all stateless offloads, switch filtering and the classification filters. [Table 7-27](#) lists which fields of the MPLS header are digested by the device. [Figure 7-12](#) shows the structure of the MPLS header.

7.3.1.4.1 Identification of Next Protocol

The X710-TM4 supports IP after MPLS by evaluating the nibble (4 bits) immediately following the last MPLS tag.

- If it equals to 0x4, the next protocol is identified as IPv4.
- If it equals to 0x6, the next protocol is identified as IPv6.
- Else, the packet is handled as L2 packet.

Table 7-27. MPLS Header Structure

Bit Offset	Size [bits]	Field	Value	Action	Comment
0	20	MPLS Label	Variable	Ignore	
20	3	QoS and ECN	Variable	Ignore	
23	1	S (bottom of stack)	0 / 1	Check	Identify the last MPLS header when 'S' = 1b.
24	8	TTL	Variable	Ignore	

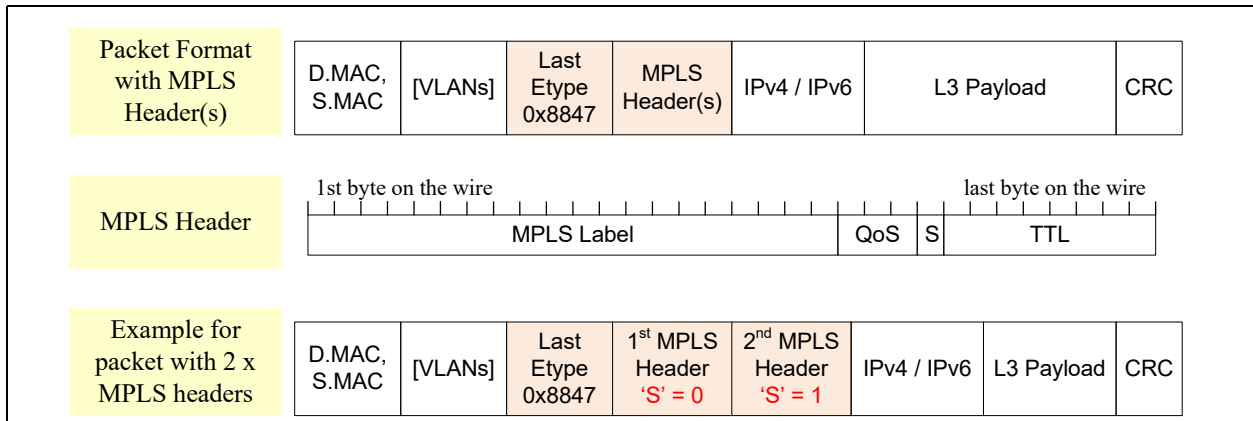


Figure 7-12. MPLS Header

7.3.1.5 LLDP Frame Format

The general structure of an LLDP packet is listed in Table 7-28.

Table 7-28. Structure of LLDP Frame

LLDP Header	
LLDP Ethertype	LLDPDU

The LLDPDU contains an ordered sequence of three mandatory TLVs followed by zero or more optional TLVs plus an end of LLDPDU TLV, as listed in Table 7-29.

Table 7-29. Structure of LLDP PDU

Mandatory	Mandatory	Mandatory					Mandatory
Chassis ID TLV	Port ID TLV	Time to Live TLV	Optional TLV	...	Optional TLV	End of LLDPDU TLV	

Each TLV has the structure listed in Table 7-30.

Table 7-30. Structure of LLDP TLV

TLV Header		
7 bits	9 bits	0 <= n <= 511 octets
TLV type	TLV information string length	TLV information string

The basic format for organizationally-specific TLVs is listed in Table 7-31.



Table 7-31. Structure of LLDP Organizationally-specific TLV

TLV Header		TLV information string - 0 ≤ n ≤ 511 octets		
7 bits	9 bits	3 octets	1 octet	0 ≤ n ≤ 507 octets
TLV type	TLV information string length	Organizationally Unique Identifier (OUI)	Organizationally-defined subtype	

Table 7-32 lists the applicable organizationally-specific TLVs.

Table 7-32. List of Organizationally-specific TLVs

TLV	OUI Value	Subtype Value
DCBx	00-80-C2	09, 0A, 0B, 0C
EEE	00-12-0F	05

7.3.1.6 Other L2 Packets

Magic packets are described in Section 5.4.1.

Link control packets are described in Section 3.2.1.5.1. Such packets do not include any L2 tags other than those described in Section 3.2.1.5.1.

7.3.2 L3 Packet Formats

7.3.2.1 IPv4 Datagram

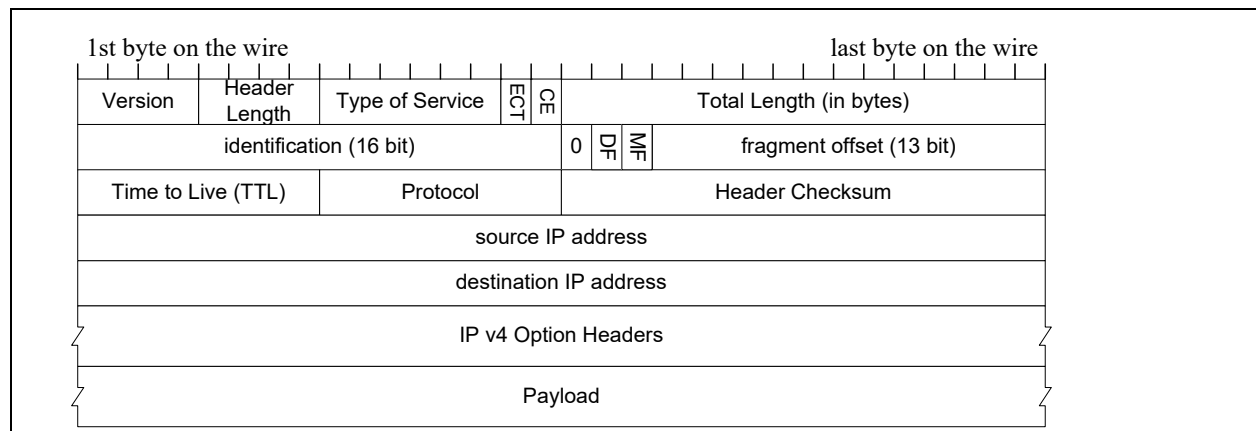


Figure 7-13. IPv4 Datagram

Version — The *Version* field is set to 0x4 for IPv4 header.



Header Length — The length of the IP header defined in Dword units.

Type of Service (TOS); 6 bits — The *TOS* field is used to indicate the quality of service with which this datagram is to be delivered by the internetwork routers.

IP Congestion Flags — CE, etc.

Total Length — The total length is the size of the IP datagram (IP header and payload) in byte units.

Identification (ID) — The *Identification* field identifies a specific IP packet sent between a source and destination node. The sending host sets the *Identification* field's value, and the field is incremented for successive IP datagrams. The *Identification* field is used to identify multiple fragments of an original IP datagram.

Fragment Parameters — Fragment offset, *More Fragment(s)* flag and *Disable Fragmentation* flag.

Time to Live (TTL) — The *TTL* field indicates the number of links that this IP datagram can travel before an IP router discards it.

Protocol (Next Header) — The *Protocol* field indicates the next protocol encapsulated within the IP layer. See [Section 7.3.2.3](#) for the list those protocols that can be offloaded by the X710-TM4.

Header Checksum — The *Checksum* field is a 16-bit one's complement of the one's complement sum of all 16-bit words in the IP header.

Source and Destination IP Addresses — 2 x 32-bit IP addresses.

7.3.2.2 IPv6 Datagram

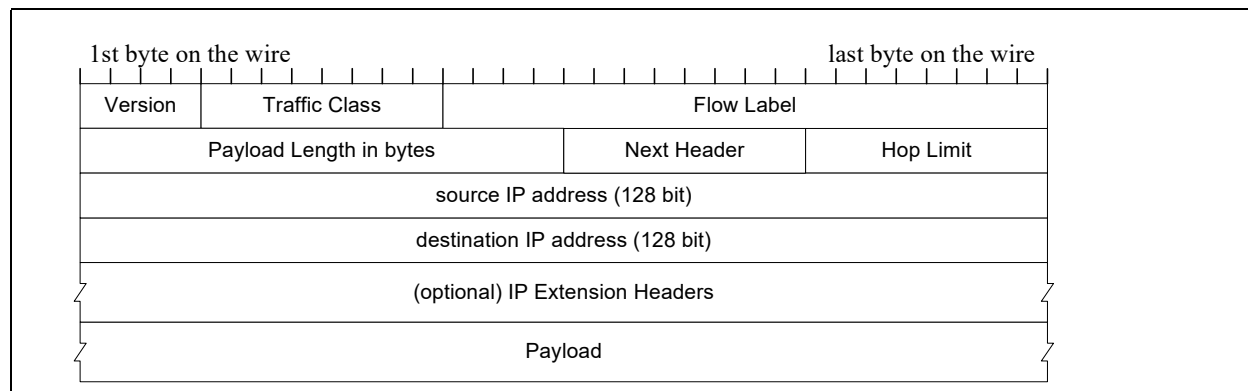


Figure 7-14. IPv6 Datagram

- Version — The *Version* field is set to 0x6 for IPv6 header.
- Traffic Class and Flow Label — The traffic class and flow label are used for QoS support.
- Payload Length — The length of the IPv6 payload. For example, the rest of the packet following the IPv6 header in byte units. Note that any IPv6 extension headers are considered part of the payload.
- Next Header (Protocol) — The *Next Header* field indicates the next protocol encapsulated within the IP layer. See [Section 7.3.2.3](#) for the list those protocols that can be offloaded by the X710-TM4.
- Hop Limit (TTL) — The hop limit indicates the number of links that this IP datagram can travel before an IP router discards it.
- Source and Destination IP Addresses — 2 x 128-bit IP addresses.

7.3.2.2.1 IPv6 Extension Headers

The IPv4 option headers that are included as part of the IP header are replaced in IPv6 by separate extension headers per option. Table 7-33 lists those most used IPv6 extension and their recommended ordering in the packet. Then these extension headers are illustrated in the following figures. The X710-TM4 identifies these headers and skip them parsing the rest of the packet for its processing (unless specified differently in the following table).

Table 7-33. IPv6 Extension Headers/Recommended Ordering

Header (Protocol)	Next Header Value	Header Length/Header Length Field Offset
Hop-by-Hop Options	0	Variable length field defined in 8-byte units excluding the first 8 bytes.
Destination Options	60	Variable length field defined in 8-byte units excluding the first 8 bytes. Can be located either at this location or before the mobility header.
Routing Header	43	Variable length field defined in 8-byte units excluding the first 8 bytes.
Fragment Header	44	Length is always 8 bytes. The X710-TM4 does not continue to parse the rest of the packet.
Authentication Header	51	Length field is at byte 1. Defines the header length minus 2 in 4-byte units. When this header is found, the X710-TM4 does not continue to parse the rest of the packet. Applicable for IPv4 as well.
Encapsulating Security Payload	50	Length is 8 bytes plus variable length of initial value plus a trailer. When this header is found, the X710-TM4 does not continue to parse the rest of the packet. Applicable to IPv4 as well.
Destination Options	60	Variable length field defined in 8-byte units excluding the first 8 bytes. Can be located either at this location or before the routing header.
Mobility Header	135	Variable length field defined in 8-byte units excluding the first 8 bytes.
No Next Header	59	When no next header type is found, the rest of the packet is not processed.

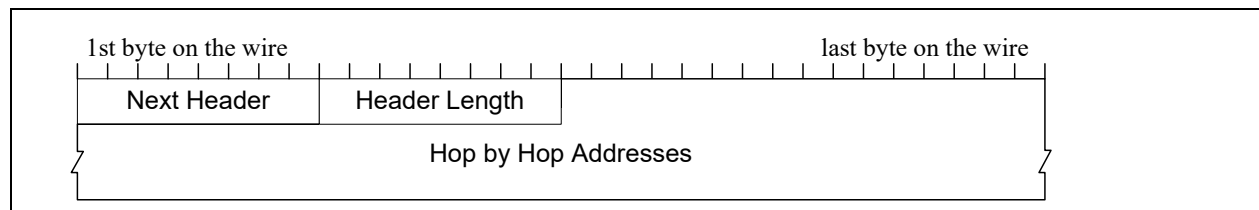


Figure 7-15. IPv6 Hop-by-Hop Extension Header

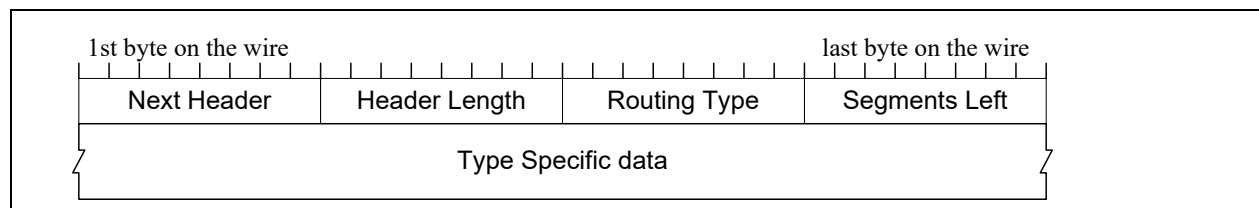


Figure 7-16. IPv6 Routing Header

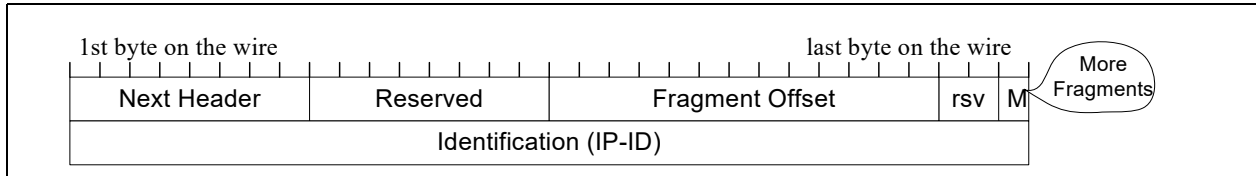


Figure 7-17. IPv6 Fragment Header

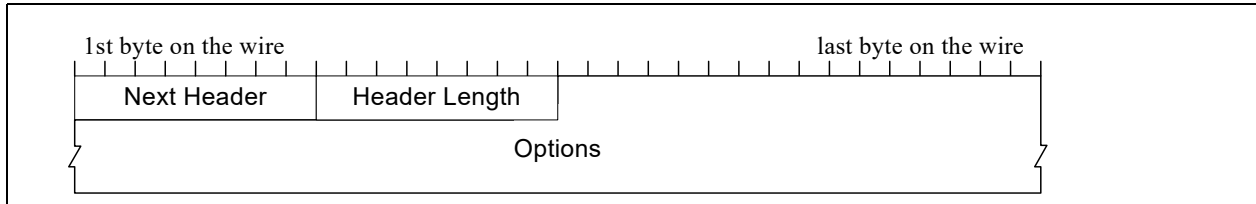


Figure 7-18. IPv6 Destination Options

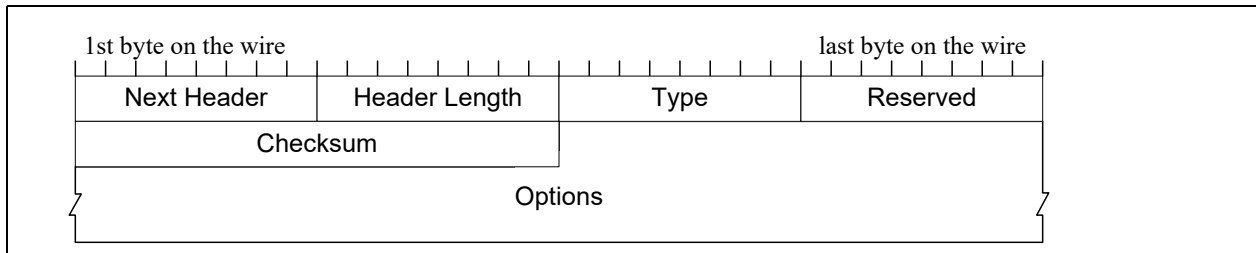


Figure 7-19. IPv6 Mobility Header

7.3.2.3 Protocol Headers (Next Header)

Table 7-34 lists the encoding of the *Next Header Type* field and information on determining each header type's length recognized by the X710-TM4 according to the recommended ordering.

Table 7-34. Header Type Encoding and Lengths

Header (Protocol)	Next Header Value	Header Length and Header Length Field Offset
IPv4	4	<i>Length</i> Field is at bits[7:4] and defined in 4-byte units.
IPv6	41	Length is always 40 bytes.
Authentication	51	<i>Length</i> Field is at byte 1 and defined by the header length minus 2 in 4-byte units.
Encapsulating Security Payload	50	Length is 8 bytes plus the variable length of the initial value plus a trailer.
TCP	6	<i>Length</i> Field is at byte 12, bits [7:4] and defined in 4-byte units.
UDP	17	Length is always 8 bytes.
ICMP	1	Length is always 8 bytes.
ICMPv6	58	Length is always 4 bytes.
SCTP	132	Length is always 12 bytes.
No Next Header	59	When no next header type is found the remainder of the packet is not processed.



7.3.2.4 ICMP Datagram

ICMP packets are used as part of the manageability filtering.

Table 7-35. ICMP Packet Format

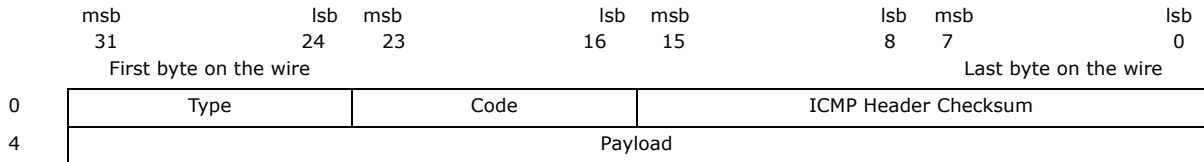


Table 7-36 lists the processing done to identify ICMP packets.

Table 7-36. IPv4 Packet Structure and Processing

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC Header Processed by the main address filter.
6	6	Source Address		Skip	
12	T=(0/4)	Possible S-tag		Skip	
12 + T	S=(0/4)	Possible VLAN Tag		Compare	Processed by the main address filter.
12 + T + S	2	Type	0x0800	Compare	IPv4
14 + T + S	1	Version/ HDR Length	0x4X	Compare	Check IPv4
15 + T + S	1	Type of Service	-	Ignore	
16 + T + S	2	Packet Length	-	Ignore	
18 + T + S	2	Identification	-	Ignore	
20 + T + S	2	Fragment Information	-	Ignore	
22 + T + S	1	Time to Live	-	Ignore	
23 + T + S	1	Protocol	0x1	Compare	ICMP
24 + T + S	2	Header Checksum	-	Ignore	
26 + T + S	4	Source IP Address	-	Ignore	
30 + T + S	4	Destination IP Address	-	Ignore	
34 + T + S	1	ICMP Type	-	Ignore	
35 + T + S	1	ICMP Code	-	Ignore	
36 + T + S	2	ICMP Header Checksum	-	Ignore	
38 + T + S	F	ICMP Payload	-	Ignore	

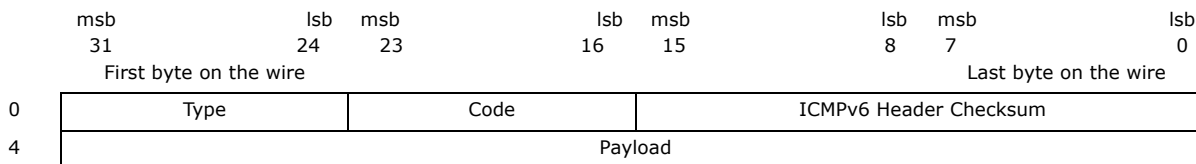
The X710-TM4 does not parse the ICMP header and only detects the IP next protocol as ICMP.



7.3.2.5 ICMPv6 Datagram

ICMP packets are used as part of the manageability filtering and as part of proxying capabilities.

Table 7-37. ICMPv6 Packet Format



The X710-TM4 supports filtering of the following ICMPv6 packets.

Neighbor discovery packets:

1. 0x86 (134d) — Router Advertisement.
2. 0x87 (135d) — Neighbor Solicitation.
3. 0x88 (136d) — Neighbor Advertisement.
4. 0x89 (137d) — Redirect.

MLD packets:

1. 0x82 (130d) — MLD Query.
2. 0x83 (131d) — MLDv1 Report.
3. 0x84 (132d) — MLD Done.
4. 0x8F (143d) — MLDv2 Report.

Table 7-38 lists the processing done to identify ICMPv6 packets.

Table 7-38. ICMPv6 Packet Structure and Processing

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC Header. Processed by the main address filter.
6	6	Source Address		Skip	
12	T=(0/4/8)	Possible S-tag		Skip	
12 + T	S=(0/4)	Possible VLAN Tag		Compare	Processed by the main address filter.
12 + T + S +	2	Type	0x86DD	Compare	IPv6
14 + T + S	1	Version/ Priority	0x6X	Compare	Check IPv6
15 + T + S	3	Flow Label		Ignore	
18 + T + S	2	Payload Length		Ignore	
20 + T + S	1	Next Header	IPv6 next header types or 0x3A	Compare	0x3A. ICMPv6 header type.
21 + T + S	1	Hop Limit	0x1	Ignore	
22 + T + S	16	Source IP Address		Ignore	



Table 7-38. ICMPv6 Packet Structure and Processing

Offset	# of Bytes	Field	Value	Action	Comment
38 + T + S	16	Destination IP Address		Ignore	
54 + T + S	N	Possible IPv6 Next Headers		Ignore	
ICMPv6 Header					
54 + T + S + N	1	Type	0x82, 0x83, 0x86, 0x87,0x88, 0x89 or 0x8F	Compare	Multicast Listener Discovery (MLD) or neighbor discovery types.
55 + T + S + N	1	Code	0x0	Ignore	
56 + T + S + N	2	Checksum		Check	
58 + T + S + N	F	Message Body		Ignore	

7.3.2.6 ARP Packets

ARP packets are used as part of manageability filtering.

Table 7-39 lists and Figure 7-20 shows the ARP packets format and the processing of these packets.

Table 7-39. ARP Packet Structure and Processing

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC Header. Processed by the main address filter.
6	6	Source Address		Skip	
12	T=(0/4/8)	Possible S-tag		Skip	
12 + T	S=(0/4)	Possible VLAN Tag		Compare	Processed by the main address filter.
12+ T + S	2	Type	0x0806	Compare	ARP
14 + T + S	2	Hardware Type	0x0001	Compare	Ethernet hardware type.
16 + T + S	2	Protocol Type	0x0800	Compare	IPv4 protocol.
18 + T + S	1	Hardware Size	0x06	Compare	MAC address size in bytes.
19 + T + S	1	Protocol Address Length	0x04	Compare	IPv4 address size in bytes.
20 + T + S	2	Operation	0x0001/ 0x0002	Compare	0x0001 = Request. 0x0002 = Response.
22 + T + S	6	Sender Hardware Address	-	Ignore	
28+ T + S	4	Sender IP Address	-	Ignore	
32 + T + S	6	Target Hardware Address	-	Ignore	
38 + T + S	4	Target IP Address	IP4AT	Compare	Use to decide of an IP match of ARP packets.

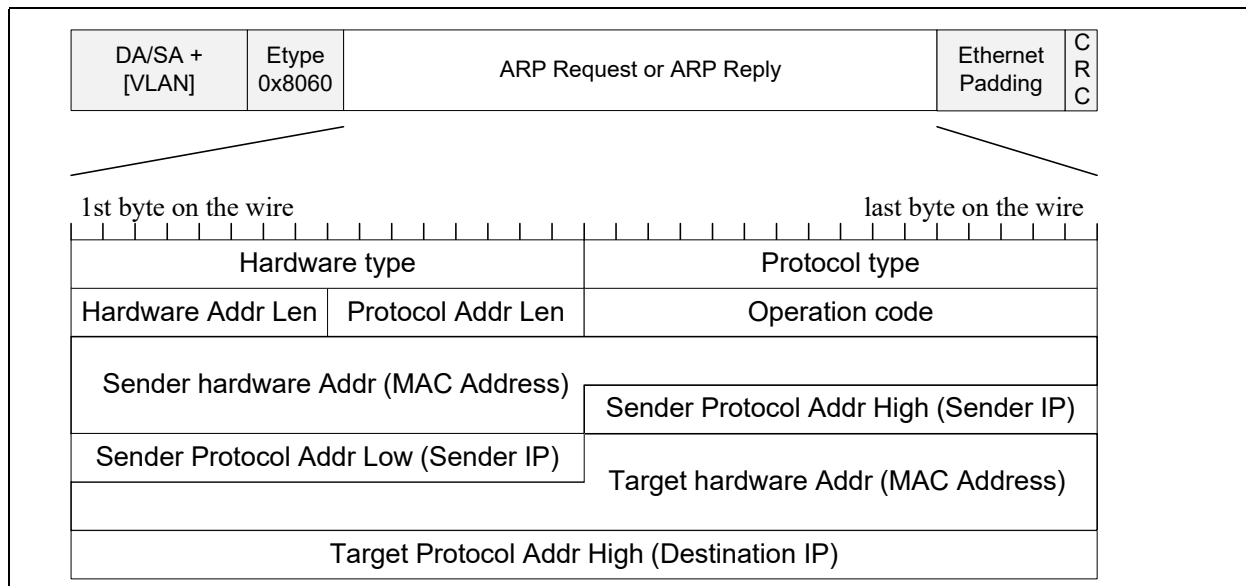


Figure 7-20. ARP Packet Format

7.3.3 L4 Packet Formats

7.3.3.1 UDP Datagram

- Source and Destination Port Number — 16-bit port numbers (see [Figure 7-21](#)).
- UDP Length — The length of the entire UDP datagram, including both *Header* and *Data* fields defined in byte units.
- UDP Checksum — An optional one's complement of the one's complement sum of all 16-bit words in the header and payload and a pseudo header shown in [Figure 7-22](#) and [Figure 7-23](#). On the transmit flow, the operating system stack provides the pseudo header checksum in the UDP checksum field when requesting from the device to offload the checksum calculation.

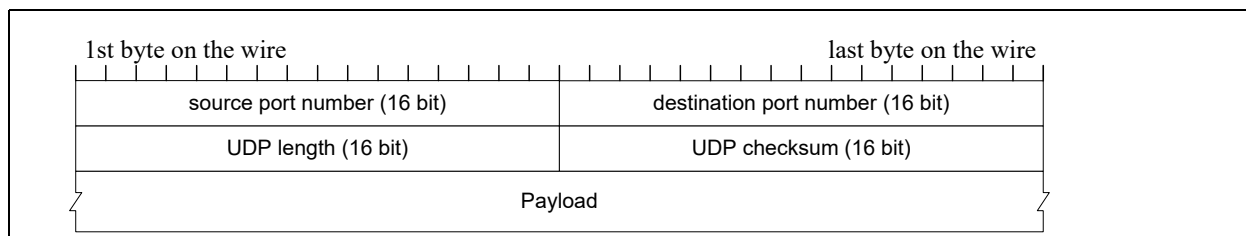


Figure 7-21. UDP Packet Format

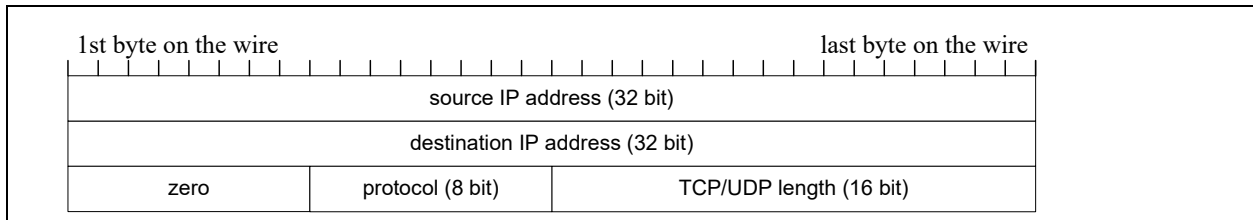


Figure 7-22. IPv4 Pseudo Header for TCP/UDP Checksum

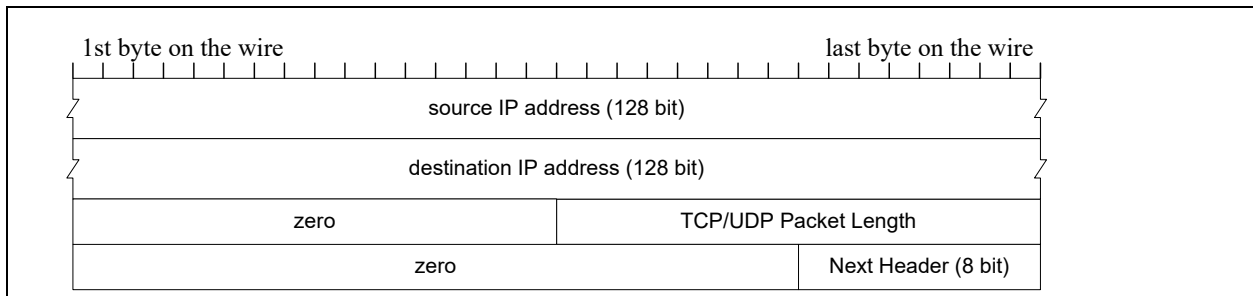


Figure 7-23. IPv6 Pseudo Header for TCP/UDP Checksum

7.3.3.2 TCP Datagram

- Source and Destination Port Number — 16-bit port numbers.
- Sequence Number — The sequence number of the first data octet in this segment (except when SYN is present). If SYN is present, the sequence number is the Initial Sequence Number (ISN) and the first data octet is ISN+1.
- Acknowledge Sequence Number — If the ACK control bit is set, this field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established this is always sent.
- Window Size — The number of data octets beginning with the one indicated in the acknowledgment field that the sender of this segment is willing to accept.
- TCP Flags (9 bits) — FIN, SYN, RST, PSH, ACK, URG, ECE, CWR and NS.
- Header Length (4-bit) — The number of Dwords in the TCP header. This indicates where the data begins. The TCP header (including TCP options) is always an integral number of 32 bits long.
- Urgent Pointer — This field communicates the current value of the urgent pointer as a positive offset from the sequence number in this segment. The urgent pointer points to the sequence number of the octet following the urgent data. This field is only interpreted in segments with the URG control bit set.
- TCP Checksum — The checksum field is the 16-bit one's complement of the one's complement sum of all 16-bit words in the header and payload and the pseudo header shown in the [Figure 7-22](#) and [Figure 7-23](#). On the transmit flow, the operating system stack provides the pseudo header checksum in the TCP checksum field when requesting from the device to offload the checksum calculation.

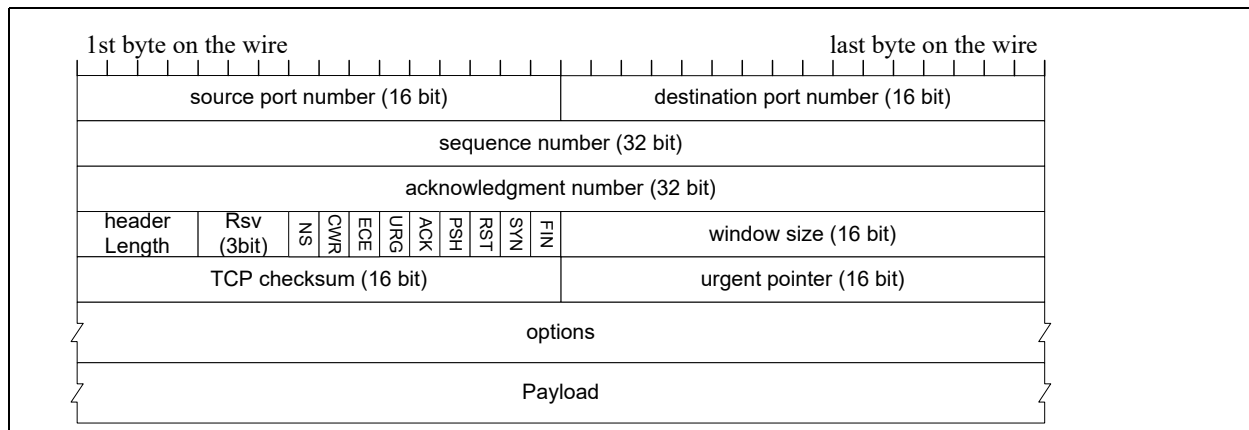


Figure 7-24. TCP Packet Format

7.3.3.3 SCTP Datagram

- Source and Destination Port Number — 16-bit port numbers.
- Verification Tag — 32-bit random value selected by each endpoint in an association during setup. It is used to discriminate between two successive associations as well as a protection mechanism against blind attackers.
- CRC Integrity — CRC32c integrity checksum covering the entire SCTP packet (SCTP header and all chunks). the CRC32c is the same polynomial used for iSCSI as follows: $1 + x + x^2 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11} + x^{12} + x^{16} + x^{22} + x^{23} + x^{26} + x^{32}$. The CRC bytes are transmitted on the network in big Endian ordering while the MS bytes are first on the wire.

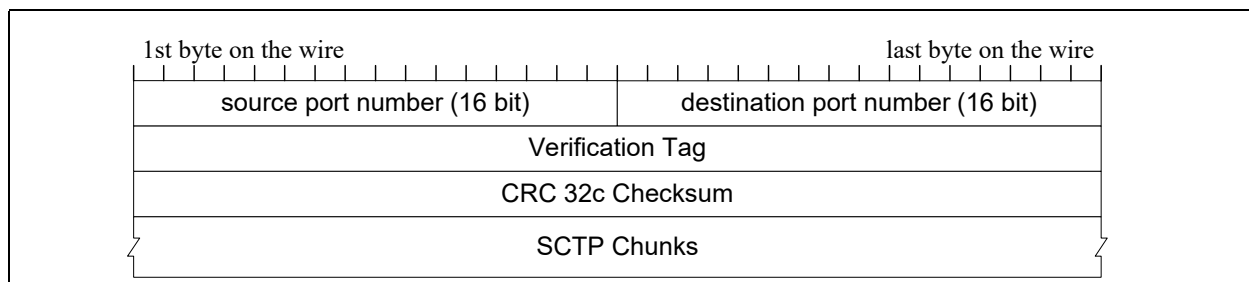


Figure 7-25. SCTP Packet Format

7.3.4 Supported Tunneled Packet Formats

Figure 7-26 shows the supported tunneled packet formats.

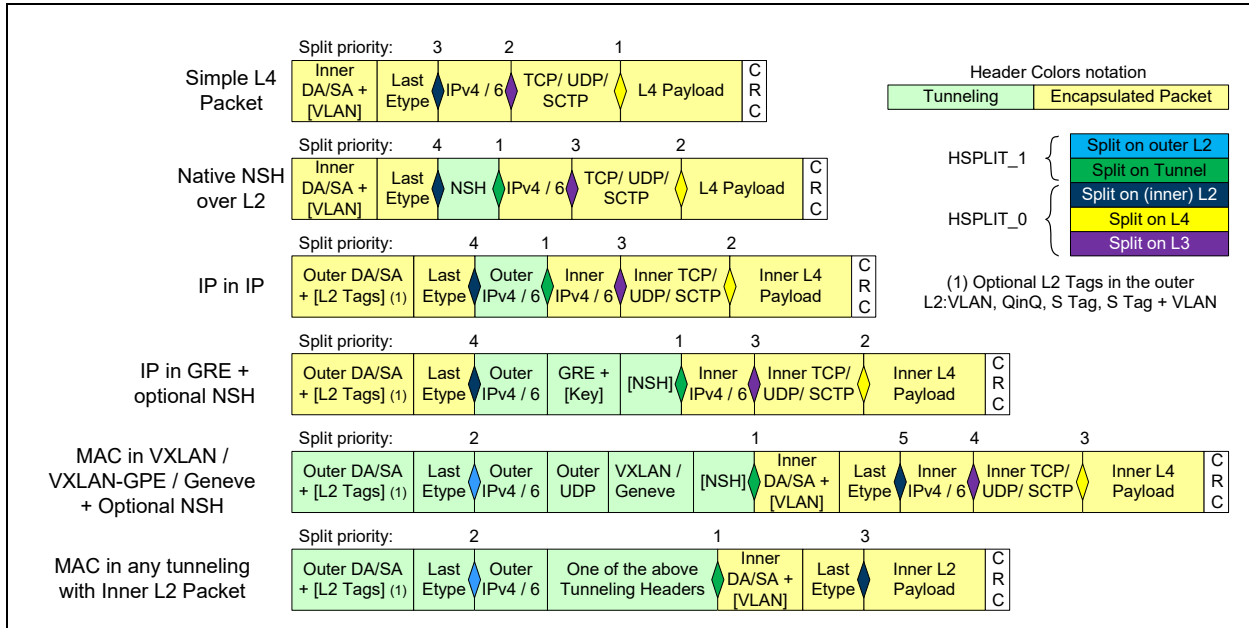


Figure 7-26. Tunneled Packet Formats

Processing of these packets is described in the LAN Engine, Internal Switch and Receive Classification Filters sections.

7.3.4.1 IP and UDP Tunneling

The X710-TM4 supports the following tunneled packets:

- IPv4 - IPv4
- IPv4 - IPv6
- IPv6 - IPv4
- IPv6 - IPv6
- IP - UDP Teredo. This IP can be IPv4 or IPv6 as previously described.

The IP headers might have options (IPv4) or extended headers (IPv6) described in Section 7.3.2.2.1.

UDP Teredo header is identified by its destination port number equals to one of 16 values as detailed in Section 7.1.7.1. The X710-TM4 does not provide checksum offload for the Teredo header with no reporting.

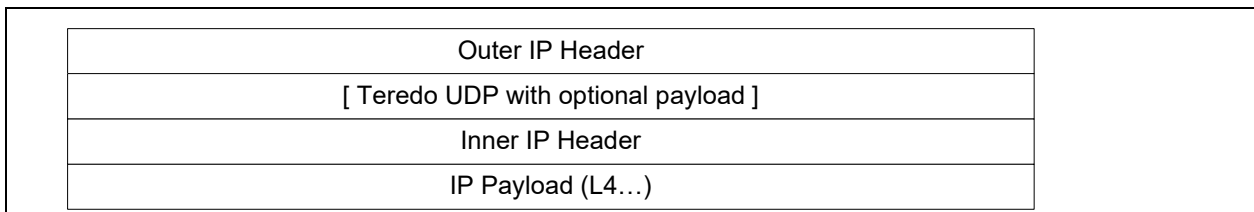


Figure 7-27. IP / Teredo Tunneling



7.3.4.2 GRE Header Version 0 (NVGRE)

1st Byte on the Wire							last Byte on the Wire		
C	R	K	S	s	Recur	Flags	Version	Protocol Type	
Checksum (Optional)							Offset (Optional)		
Key (Optional) - Tenant Network ID (TNI)							Reserved		
Sequence Number (Optional)									
Routing ::: (Optional)									

The GRE header is indicated by the IP protocol that equals to 47 (0x2F). The GRE headers supported by the X710-TM4 can be 1, 2, 3 or 4 Dwords depending on the flags in the first byte.

- 'C' (Checksum Present) — When set, it indicates that the *Checksum* field is present and contains valid information. If either the *Checksum Present* bit or the *Routing Present* bit are set, the *Checksum* and *Offset* fields are both present.
- 'R' (Routing Present) — GRE header with routing header is not supported by the X710-TM4. If this flag is found active, the header is not recognized by the device.
- 'K' (Key Present) — If set, the *Key* field is present and contains valid information.
- 'S' (Sequence Number) — If set, the *Sequence Number* field is present and contains valid information. The X710-TM4 ignores this flag other than an indication for the length of this header.
- 's' (Strict Source Route) — It is recommended that this bit only be set if all of the routing information consists of strict source routes. The X710-TM4 ignores this flag.
- 'Recur' (3 bits) - Recursion Control — Contains the number of additional encapsulations that are permitted. The X710-TM4 supports only GRE header with no recursion headers (recursion control equals to zero).
- 'Version' (3 bits) — GRE protocol version. Zero value identifies a GRE header version zero. The X710-TM4 supports only zero value. (Note that version 1 is used for PPP protocol).
- 'Protocol' (16 bits) — Contains the protocol type of the payload packet. In general, the value is the Ethernet protocol type field for the packet. The following values are supported by the X710-TM4. Packet parsing that might have other protocol values is undefined.
 - 0x0800: IPv4 header (indicates an IPv4 in GRE packet format)
 - 0x86DD: IPv6 header (indicates an IPv6 in GRE packet format)
 - 0x6558: MAC header (indicates a MAC in GRE packet format)
 - 0x894F: NSH header (defined by the NSH *Protocol* field)
- 'Checksum' (16 bits) — Contains the IP (one's complement) checksum of the GRE header and the payload packet. This field is not processed by the X710-TM4.
- 'Offset' (16 bits) — Indicates the byte offset from the start of the *Routing* field to the first byte of the active source route entry to be examined. This field is not processed by the X710-TM4.
- 'Key' (24 bits) — Contains a number that was inserted by the encapsulator. It might be used by the receiver to authenticate the source of the packet. The GRE key is used for VSI classification if enabled by the switch filters.
- 'Sequence Number' (32 bits) — Contains a number that is inserted by the encapsulator. It might be used by the receiver to establish the order in which packets have been transmitted from the encapsulator to the receiver. This field is not processed by the X710-TM4.
- 'Routing' (Variable Length) — This field is a list of SREs and is not supported by the X710-TM4.



7.3.4.3 UDP and its Optional Private Header



- UDP Header — The UDP destination port number equals to one of 16 Teredo port numbers (also used for MAC in UDP tunneling).
- Private Header — The private UDP header includes any optional private content that could include a networking key. The length of the private header and the optional networking key are recognized by the device according to the UDP port number as programmed by the Add Teredo Port admin command described in [Section 7.1.7.1](#).

7.3.4.3.1 VXLAN Header in UDP Format

VXLAN draft provides the structure of the optional UDP private header previously described used for VXLAN encapsulation as listed in [Table 7-40](#) and shown in [Figure 7-28](#).

Table 7-40. VXLAN Header Structure

Offset (from the UDP header)	Size [byte]	Field	Value	Action	Comment
0	2	Source UDP Port	XXXX	Ignore	Source UDP port number could be any value.
2	2	Destination UDP Port	4789	Compare	The reserved VXLAN port number is programmed by the Add Tunneling UDP admin command with Tunneling UDP Protocol Type being equal to VXLAN.
4	2	UDP Length	XXXX	Ignore	Length of the entire datagram including the UDP header.
6	2	UDP Checksum	XXXX	Ignore	Provide an indication if the checksum equals to zero (such as no checksum).
8	1	Flags	0x08	Ignore	VV = Version (expect zero). I = VNI valid indication.
9	3	Reserved	XXXX	Ignore	Reserved.
12	3	VXLAN Network Identifier	XXXX	Compare	Unique value per tenant.
15	1	Reserved	XX	Ignore	Reserved.

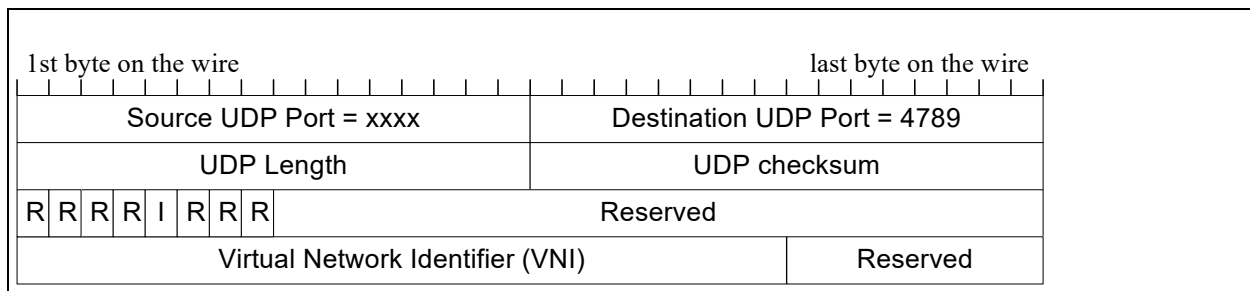


Figure 7-28. VXLAN Header Structure



7.3.4.3.2 Geneve Header in UDP Format

Geneve draft (Geneve: Generic Network Virtualization Encapsulation draft-gross-geneve-02) provides the structure of the optional UDP private header previously described used for Geneve encapsulation as listed in Table 7-41 and shown in Figure 7-29.

Table 7-41. Geneve Header Structure

Offset (from the UDP header)	Size [byte]	Field	Value	Action	Comment
0	2	Source UDP Port	XXXX	Ignore	Source UDP port number could be any value.
2	2	Destination UDP Port	6081	Compare	The reserved Geneve port number is programmed by the Add Tunneling UDP admin command with Tunneling UDP Protocol Type being equal to Geneve.
4	2	UDP Length	XXXX	Ignore	Length of the entire datagram including the UDP header.
6	2	UDP Checksum	XXXX	Ignore	
8.7 - 8.6	2 bits	V V (Version)	00b	Check ¹	
8.5 - 8.0	6 bit	Opt Len (Option Length)	Variable	Check ¹	Defines the length of the options fields in 4-byte units.
9.7	1 bit	O (OAM Frame)	0b	Check	Rout the packet to queue zero of the VSI.
9.6	1 bit	C (Critical Options Present)	X	Ignore	Expected to be processed by the software stack.
9.5 - 9.0	6 bit	Reserved	X	Ignore	
10	2	Next Protocol		Check	Next protocol that follows the Geneve header. The X710-TM4 supports the following values: 0x6558: MAC header (MAC in Geneve) 0x0800: IPv4 header (IPv4 in Geneve) 0x86DD: IPv6 header (IPv6 in Geneve) 0x894F: NSH header (NSH in Geneve) If other values are found, the packet is identified as Geneve, Payload.
12	3	VNI (Network Identifier)	Variable	Compare	Unique value per tenant.
15	1	Reserved	X	Ignore	
16	4 x Opt Len	Variable Length Options	XXXX	Ignore	The length of the <i>Options</i> field is defined by the <i>Opt Len</i> field in this header.

1. If the total header length is larger than 256 or the version <> 0, the packet is handled as a simple L2 packet type.

1st Byte on the Wire					Last Byte on the Wire	
Source UDP Port = xxxx				Destination UDP Port = 0x6081		
UDP Length				UDP Checksum		
V	Opt Len	O	C	Reserved		
Virtual Network Identifier (VNI)						Reserved
Variable Length Options (defined by Opt Len)						



Figure 7-29. Geneve Header Structure

7.3.4.3.3 VXLAN-GPE Header in UDP Format

VXLAN-GPE draft provides the structure of the optional UDP private header previously described used for VXLAN-GPE encapsulation as listed in Table 7-42 and shown in Figure 7-30. The highlighted fields in Red Color are those fields that are changed in VXLAN-GPE compared to the original VXLAN header.

Table 7-42. VXLAN-GPE Header Structure (Additive Fields on Top of Original VXLAN)

Byte Offset (from the UDP header)	Size	Field	Value	Action	Comment
2	2 byte	Destination UDP Port	4790	Compare	The reserved VXLAN-GPE port number is programmed by the Add Tunneling UDP admin command with Tunneling UDP Protocol Type being equal to VXLAN-GPE.
8.0	1 bit	O (OAM bit)	0 / 1	Compare	Packets with active OAM flag are identified as VXLAN-GPE-OAM PCTYPE and routed to Rx queue zero of the VSI.
8.2	1 bit	P (Next Protocol Field Valid)	0 / 1	Ignore	The <i>Next Protocol</i> field is valid only when the 'P' bit is set. The X710-TM4 ignores this flag based on the assumption that if 'P' is cleared, the <i>Next Protocol</i> field equals to zero.
8.3	1 bit	I (VNI Tag present)	1	Ignore	VNI valid indication must be set in the header.
8.5 - 8.4	2 bits	Version Field	00	Compare	Parsing of packets with non-zero <i>Version</i> field is aborted. It is indicated by abort PTYPE in the Rx descriptor and the switching decision is made by the MAC/VLAN headers.
11	1 byte	Next Protocol	XXXX	Compare	Next protocol that follows the VXLAN-GPE header. The X710-TM4 supports the following values: 0x0 or 0x3 MAC in VXLAN-GPE 0x1 IPv4 in VXLAN-GPE 0x2 IPv6 in VXLAN-GPE 0x4 NSH in VXLAN-GPE If other values are found, the packet is identified as VXLAN-GPE, Payload.
12 - 14	3 bytes	Virtual Network Identifier	Variable	Compare	Unique value per tenant.

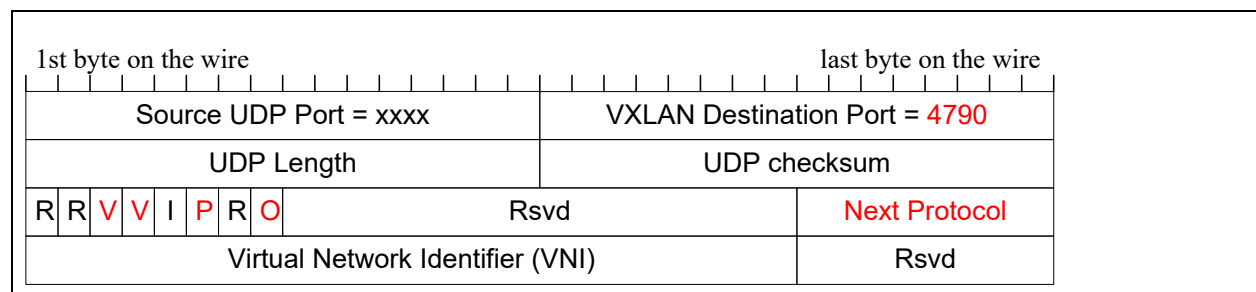


Figure 7-30. VXLAN-GPE Header Structure



7.3.4.4 NSH Header

NSH is one of the optional Next Protocol in VXLAN-GPE or in GRE or as an L2 protocol (EType = 0x894F). The structure of the NSH header is listed in Table 7-42 and shown in Figure 7-30.

Table 7-43. NSH Header Structure

Byte Offset	Size [byte]	Field	Value	Action	Comment
0.4	1 bit	C (Critical Data Flag)	0 / 1	Ignore	
0.5	1 bit	O (OAM Frame)	0 / 1	Compare	Packets with active OAM flag are identified as NSH-OAM PCTYPE and routed to Rx queue zero of the VSI.
0.7 - 0.6	2 bits	VV (NSH Version)	00b	Compare	Parsing of packets with non-zero Version field is aborted. It is indicated by Abort PTYPE in the Rx descriptor and the switching decision is made by the MAC/VLAN headers.
1.5 - 1.0	6 bits	Length	XXXX	Process	Defines the length of the NSH header in 4-byte units.
2	1	MD Type (NSH Meta Data Type)	XXXX	Ignore	
3	1	Protocol	XXXX	Process	Next protocol can be one of the following: 0x3: MAC in NSH. 0x1: IPv4 in NSH. 0x2: IPv6 in NSH. If other values are found, the packet is identified as NSH, Payload.
4 - 7	4	Service Path Header and Index	XXXX	Ignore	
8 - 23	16	Mandatory Context Header	XXXX	Ignore	
24	Variable	Optional Context Headers	XXXX	Ignore	Optional variable length context headers.

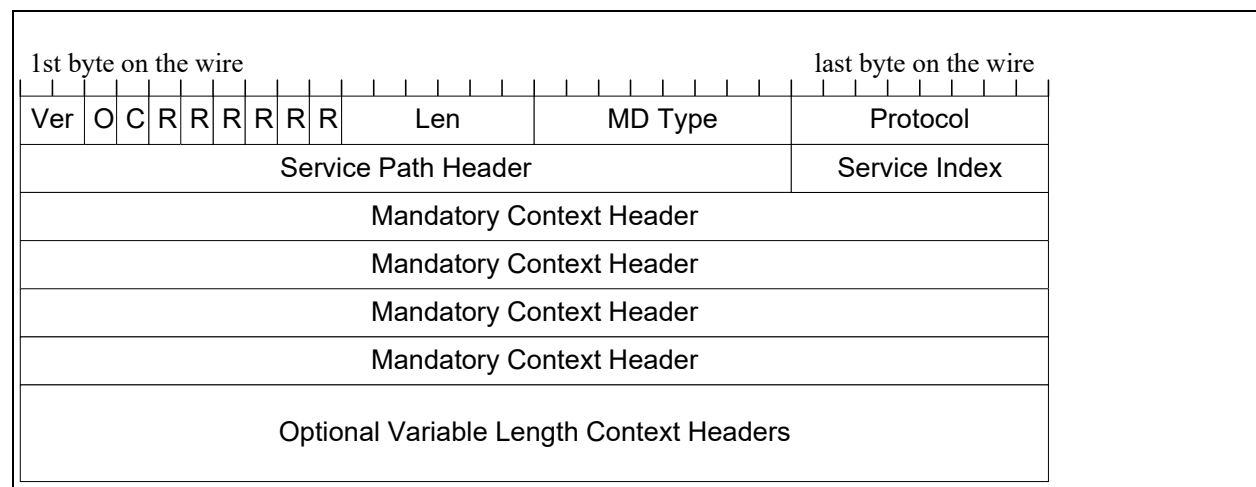


Figure 7-31. NSH Header Structure



7.3.4.4.1 NSH Header Encapsulation Options

The X710-TM4 supports the following NSH encapsulation options as shown in [Figure 7-26](#).
Native NSH over L2 (L2 Ethertype = 0x894F), NSH over GRE and NSH over VXLAN-GPE.

Note: NSH support should only be considered as NSH ready. It is enabled but not validated.



7.4 Internal Switch

7.4.1 Switch Concept

The X710-TM4 includes a number of switch elements. Part of these elements are generic L2 switching elements and part are dedicated elements that can implement a specific functionality.

These elements are compatible to the IEEE P802.1Qbg specification. [Section 7.4.2](#) describes the possible usage models of the switch as defined in these specifications. [Section 7.4.3](#) describes the management protocol used to configure the switch.

The different elements are described in [Section 7.4.4](#). Each switch can be connected to different ingress and egress ports in different configurations as described in [Section 7.4.5](#). The algorithm that defines which switch routes which packet is described in [Section 7.4.8](#).

The switches can have Virtual Station Interfaces (VSIs) connected through the PCIe interface, a physical port connected via the Ethernet interface and management ports either internal (Embedded Processor - EMP) or connected to a PCIe function. Details about the ingress and egress ports of the switch can be found in [Section 7.4.5](#).

VSIs contain a set of queues or flows and are characterized by an L2 identifier. After the packet is forwarded to a specific ingress port, an internal destination inside the port is selected as described in [Section 7.1](#) for LAN traffic.

In addition to forwarding services, the switch supports also some additional services as described in [Section 7.4.6](#).

The switch is not a learning switch and it is managed by the host. The programming interface exposed to the operating system is that of a managed switch. Each switch element might be configured either via a software device driver running on a PF or via the EMP. When a software device driver programs the switch it uses admin commands processed by the EMP as described in [Section 7.4.9](#).

The switch is built from a set of tables that can be allocated to different filtering resources and of a set of other fixed resources (VSIs, VEBs, mirroring rules, etc.). The current resource allocation and the way to allocate them to PFs are described in [Section 7.4.9.3](#).

7.4.2 Switch Configurations

X710-TM4 embedded switch can be used to offload the data plane functions of the virtual switching capability supported by many VMMs (bridging and switching are used synonymously in this document).

This section describes the logical models that can be represented by the switching elements. The actual switching elements definition can be found in [Section 7.4.4](#).

The X710-TM4 switch can be configured to operate in different modes as follows:

- Internal switching configurations — Virtual embedded bridging:
 - Software-based switching in VMM
 - Hardware-based switching for direct connected virtual machines
 - Local switching between virtual machines



- External switching configurations:
 - Virtual port aggregator where VM-to-VM switching is performed by an adjacent bridge
 - Port virtualizer configuration where the VM-to-VM switching is performed by a controlling bridge that could be in the network edge or aggregation layers.

Note: A device might work either in port aggregator or in port virtualizer mode. The mode of operation should be the same for all the ports.

- Combination of internal and external switching configurations:
 - Internal switching modes, a software switching model or direct connect model can coexist with one of these external switching modes: port aggregator or port virtualizer.

7.4.2.1 Internal Switch Configurations

In internal switching mode, the X710-TM4 switching element can be configured to offload data plane functions of the VMM-based software switches or can handle data plane switching functions for direct connected VMs (through SR-IOV or multiple PCI functions).

7.4.2.1.1 Hardware Offload of Software Virtual Bridges (VEB)

Figure 7-32 shows the hardware offload of software-based switches.

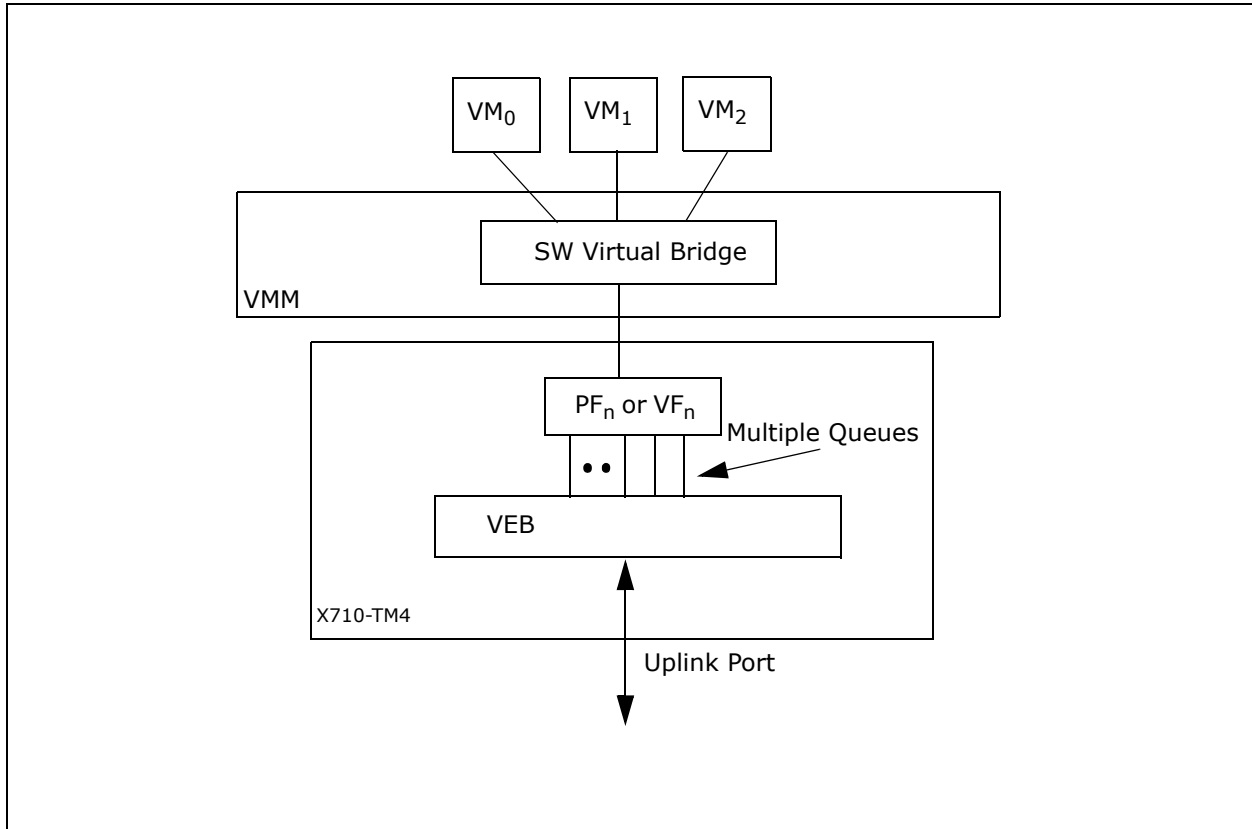


Figure 7-32. Internal Switching Configuration – Software VEB

When the X710-TM4 switching element is configured to operate in this mode, data plane functions such as the filtering and forwarding functions are offloaded to hardware. The switching element can also perform hardware replication of multicast and broadcast frames. In addition the switch can perform VLAN offload functions and security functions such as MAC address spoofing. In this mode, multiple queues are assigned to the PCI function (PF or VF for SR-IOV). Each set of queues represents a VSI in the switching element. The uplink of the switch is connected to one of the external Ethernet MAC ports.

The entire control plane functionality is handled by the software switch in the VMM via a PF. The switch hardware is programmed by a set of switch APIs. The switch control plane software manages the switching elements forwarding database, VLAN membership, bandwidth assignment or other virtual port characteristics. Since the forwarding path is still through the software intermediary layer in the VMM, it is possible for the software switch to examine the packets or perform additional functions on the frames, for example handling new protocol headers, collecting additional statistics etc.

7.4.2.1.2 VEB (Direct Connected VMs)

Figure 7-33 shows the hardware switching of direct-connected VMs.

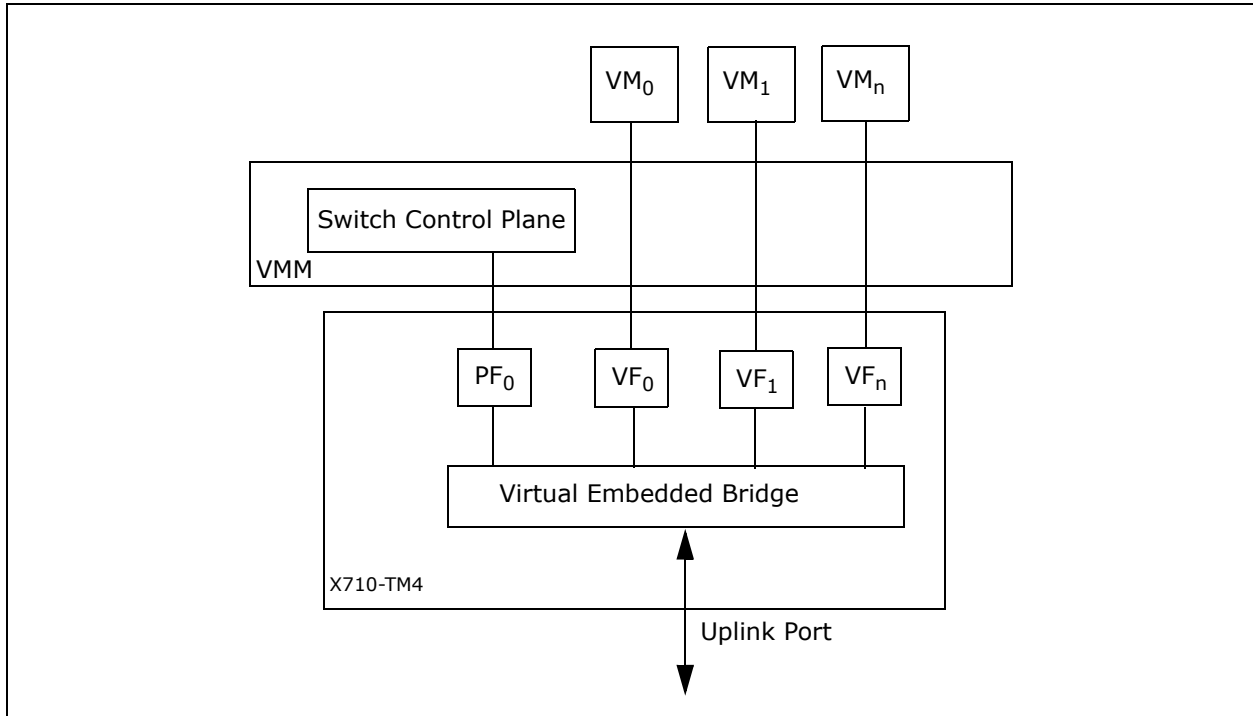


Figure 7-33. Internal Switching Configuration (Hardware VEB)

When the X710-TM4 switching element is configured to operate in this mode, each of the VSIs are directly connected to PCIe functions (PFs or VFs) that are assigned to the VMs. The software layer is eliminated in the forwarding path hence the entire data plane functionality is handled by the hardware embedded switch. All the virtual switching functions such as VLAN membership, filtering and forwarding, bandwidth management, statistics etc., are handled in hardware.

The entire control plane functionality is handled by the switch management agent in the VMM. Control packets are forwarded to the control plane agent through a management port. The switch hardware is programmed by a set of switch APIs. The switch management software manages the switching elements forwarding database, VLAN membership, bandwidth assignment or other VSI characteristics. The switch management can monitor the traffic by accessing the statistics counters or can also monitor data traffic to VM by enabling mirroring functionality in the embedded switch.

Multiple virtual embedded switching elements can be instantiated in the X710-TM4; however, there can be only one virtual switching element per uplink port. If there are multiple switching elements on a single uplink port, then a multi-channel capable S-component (See [Section 7.4.4.2.1](#)) needs to be configured on the uplink. However, a virtual switching element can be configured without an uplink port for local switching between VMs. This is used for configuring internal networks.

7.4.2.2 External Switch Configurations

The X710-TM4 supports external switching modes. In external switching modes, the virtual switching functionality including VM-to-VM, and VM-to-uplink forwarding is handled by external switches. This enables a single network management domain to manage both enterprise switches and virtual switching. The entire data plane functionality and the control plane functionality is handled by the external switch.



Two protocols are being defined by IEEE to enable virtual bridging in external switches: IEEE P802.1Qbg and IEEE P802.1BR. The X710-TM4 supports the IEEE P802.1Qbg model. The virtual port aggregator model is very similar to the embedded virtual switching model.

7.4.2.2.1 Virtual Port Aggregator Model

Figure 7-34 shows a port aggregator configuration.

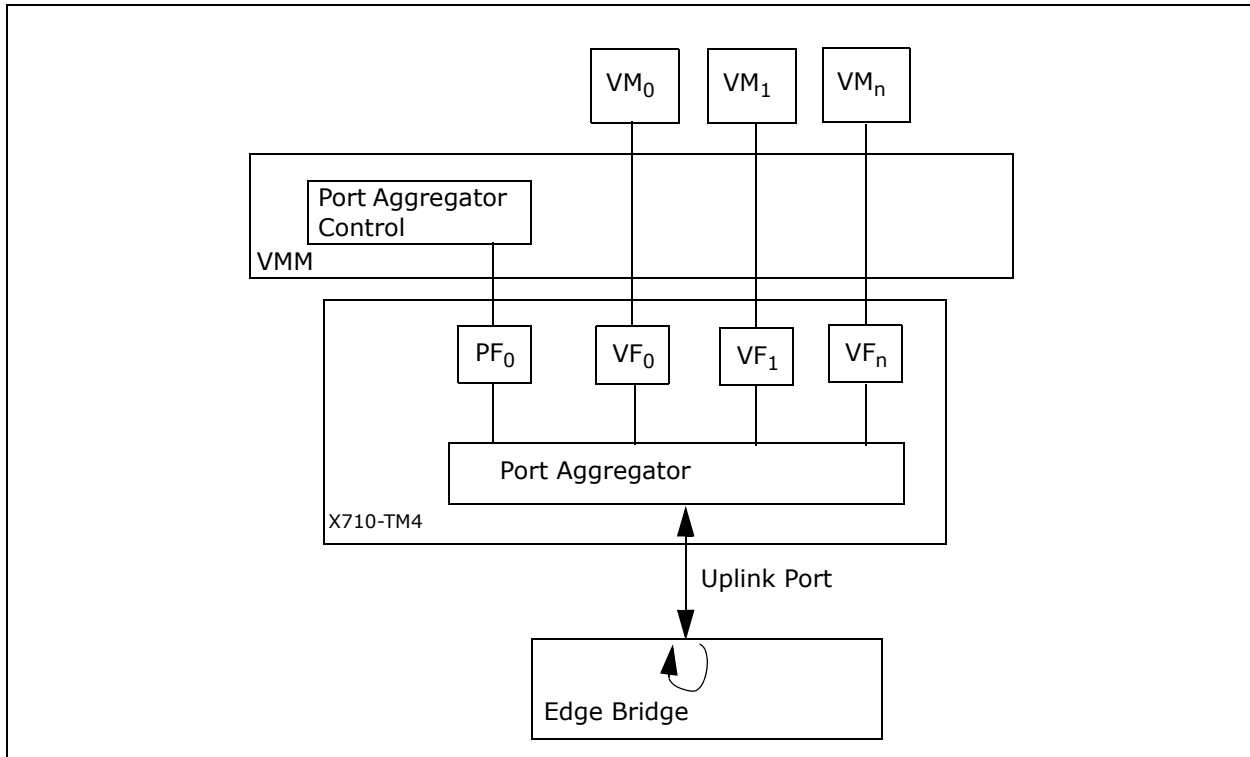


Figure 7-34. External Switching Configuration (Virtual Port Aggregator)

In a simple port aggregator mode, there is no change to the Ethernet frame format. During the ingress direction, the operation is similar to VEB, the packets are forwarded to the VMs by querying into the forwarding database in the embedded switch. In the egress direction, all traffic including the VM-to-VM traffic is forwarded to the external switch. The external switch performs lookup on the packets and turns-around the packet back to the end node (called reflexive relay service or hairpin forwarding) if the packet has to go to another VM residing in the same end station. Since all packets flow through the external switch, all advanced features of the external switch (such as advanced ACLs and flow monitoring features) are available for VM-to-VM traffic. However, the bandwidth of the uplink is shared between the VM-to-VM traffic and the uplink traffic; there is also additional latency for externally switched VM-to-VM traffic.

In port aggregator mode, hardware performs broadcast and multicast replication functions. During multicast and broadcast, the packet is forwarded by the external switch once to the end station and the port aggregator in the X710-TM4 replicates the packet multiple times to all the multicast member VMs attached to the port aggregator.

The VSIs of a port aggregator can be directly connected to the VMs through PCIe functions or VFs in SR-IOV mode. A cascaded port aggregator configuration is also allowed where a software-based port aggregator in VMM can be connected through one of the virtual ports. In this case, the port is designated as a cascaded port (See Section 7.4.2.4).

The discovery and configuration of the port aggregator mode is handled by a software port aggregator control agent residing in the VMM. The hardware switch element is managed through a set of APIs accessible to the port aggregator control agent.

7.4.2.2.2 Port Virtualizer Model

There are two ways to divide the link to channels:

- A port-mapping S-VLAN component (S-comp) defined in the IEEE 802.1Qbg specification. The S-comp uses s-tags (Ethertype = 0x88A8) to define s-channels that do not support replication. See Section 7.4.2.2.2.1 and Section 7.4.4.2.1.
- A port mapping based on MAC address. This mode is used when a function is dedicated to storage function and uses a small and exclusive set of MAC addresses. This mode supports two channels only. A storage channel and a default channel receive all the other traffic. See Section 7.4.3.

Figure 7-35 shows the hardware switch element in the X710-TM4 configured to operate in S-comp configuration.

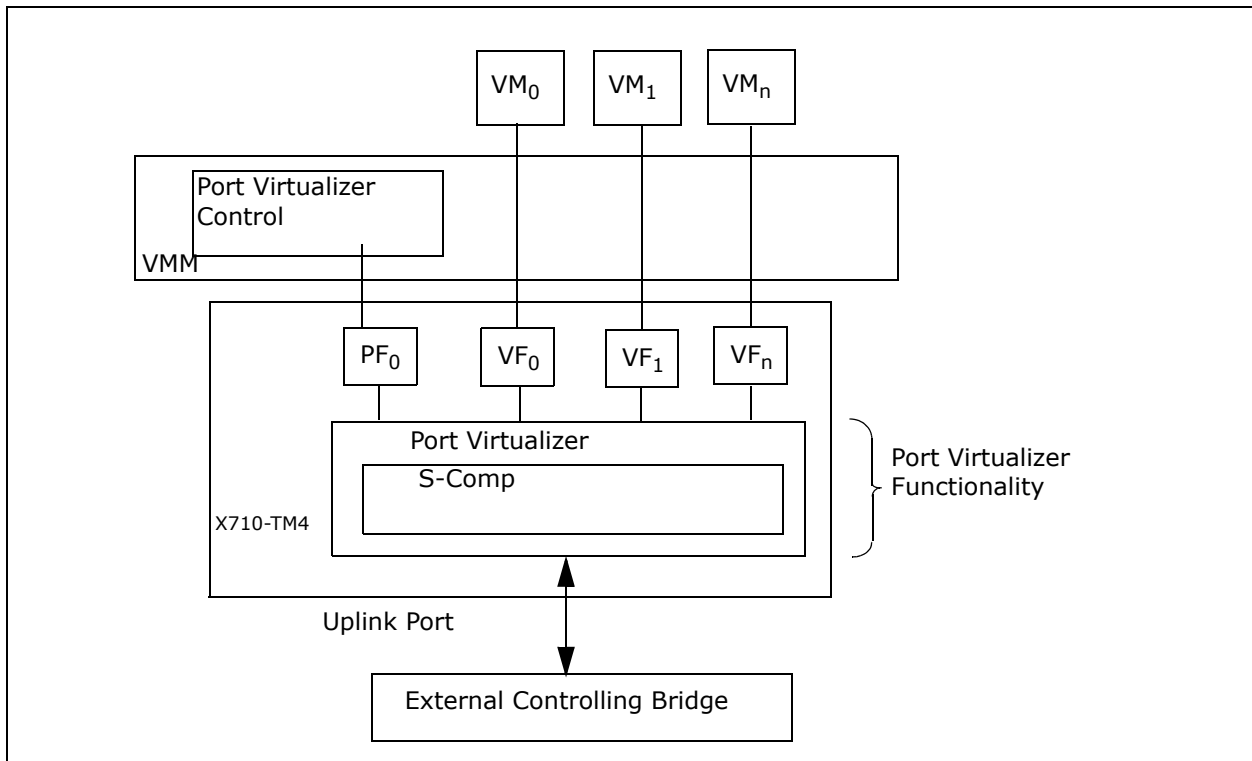


Figure 7-35. External Switching Configuration (Port Virtualizer)



7.4.2.2.2.1 S-comp Model

In an S-comp port virtualizer configuration, the controlling bridge's ports are logically extended to the virtual stations (or VMs). A physical port in the controlling bridge is configured to have multiple logical channels (or logical ports), typically one per VM. Correspondingly, the physical port in the end node is also configured to have same number of logical channels (or logical ports) that are associated with VMs. The packets need to explicitly carry a tag to identify the logical channels (or logical ports). An S-component (service VLAN component) is configured to provide this multi-channel capability. The S-tag indicates a logical channel or logical port on the controlling bridge. In this mode of operation, each VSI in the hardware switching element is associated with a logical channel in the S-component.

7.4.2.2.2.2 Host Connection

The VSIs of a port virtualizer can be directly connected to the VMs through PCIe functions or VFs in SR-IOV mode. A cascaded port virtualizer configuration is also allowed where a software-based port virtualizer in VMM can be connected through one of the virtual ports. In this case, the port is designated as a cascaded port (See [Section 7.4.2.4](#)). The discovery and configuration of the port virtualizer mode is handled by a software port virtualizer control agent residing in the VMM. The hardware switch element is managed through a set of APIs accessible to the port virtualizer control agent in VMM or IOVM.

7.4.2.2.3 SAN and LAN MFP Forwarding

[Figure 7-36](#) shows the switch structure for an MFP configuration of two functions, one for LAN and one for SAN. The switch provides the following functionality:

- The S-comp selection of S-channels is based on MAC address (see [Section 7.4.4.2.3](#)).
- No forwarding of transmit traffic between functions. A VEB might still forward transmit traffic within the LAN function.
- Receive unicast packets are forwarded to either LAN or SAN, based on destination MAC address (exact match).
- Receive multicast packets are forwarded to either LAN or SAN, based on destination MAC address (exact match).
- Receive broadcast packets are forwarded to LAN.
- Receive packets that do not match any MAC address are forwarded to the LAN.
- LAN L2 filtering might then decide to drop such a packet.
- VEB/VEPA support is expected only on the LAN function.
- Sx operation (see [Section 7.4.4.9](#)) is associated with the LAN function. Thus WoL is supported only over LAN function.

During a cold-reset sequence, the device configures a port virtualizer with two SEIDs (one per each PCI function). The LAN PF might then proceed and configure a VEB on top of its switch port.

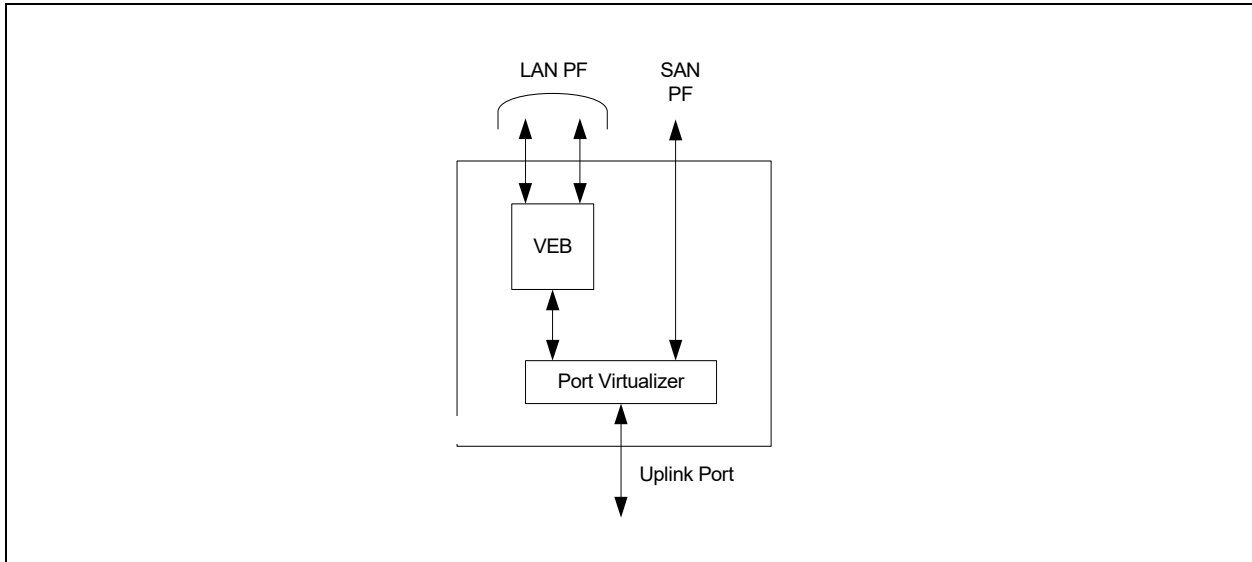


Figure 7-36. Internal Switch Configuration - LAN and SAN PFs

7.4.2.3 Coexistence of Internal/External Switch Configurations

Both internal and external switching configurations can coexist within an end station accessible through the same uplink port. Legacy VMs or applications that need higher performance for VM-to-VM traffic require VEB functionality to perform internal switching. In such usage models, the VEB coexists with either a port aggregator or port virtualizer in the same end station.

Figure 7-37 shows a combination of VEBs with a port aggregator.

In this configuration, multiple switching elements are instantiated in the X710-TM4 on the same uplink port. Each instance is accessible through a multi-channel S-component. The S-component is used to mux/demux multiple logical channels (S-channels) from the same physical port. The S-tag identifies the logical channels (S-channels) or logical ports that are extended from the adjacent bridge.

In this example configuration, the legacy VMs are connected through a software-based VEB connected through one of the S-channel ports. A second VM (that hosts a virtual appliance) is directly connected to the external switch through the second S-channel port. In addition, a port aggregator is configured through the third S-channel port that provides external switching functionality to VMs. Certain higher performance applications that require heavy VM-to-VM traffic (like a co-located front end and mid tier application servers) are configured through directly connected VFs to an embedded bridge. This embedded bridge is accessible through fourth S-channel port.

Discovery and configuration of individual switching components is performed through appropriate control agents in the VMM or IOVM.

The configuration of the switch is done using the CDCP protocol as defined in 802.1Qbg specification.

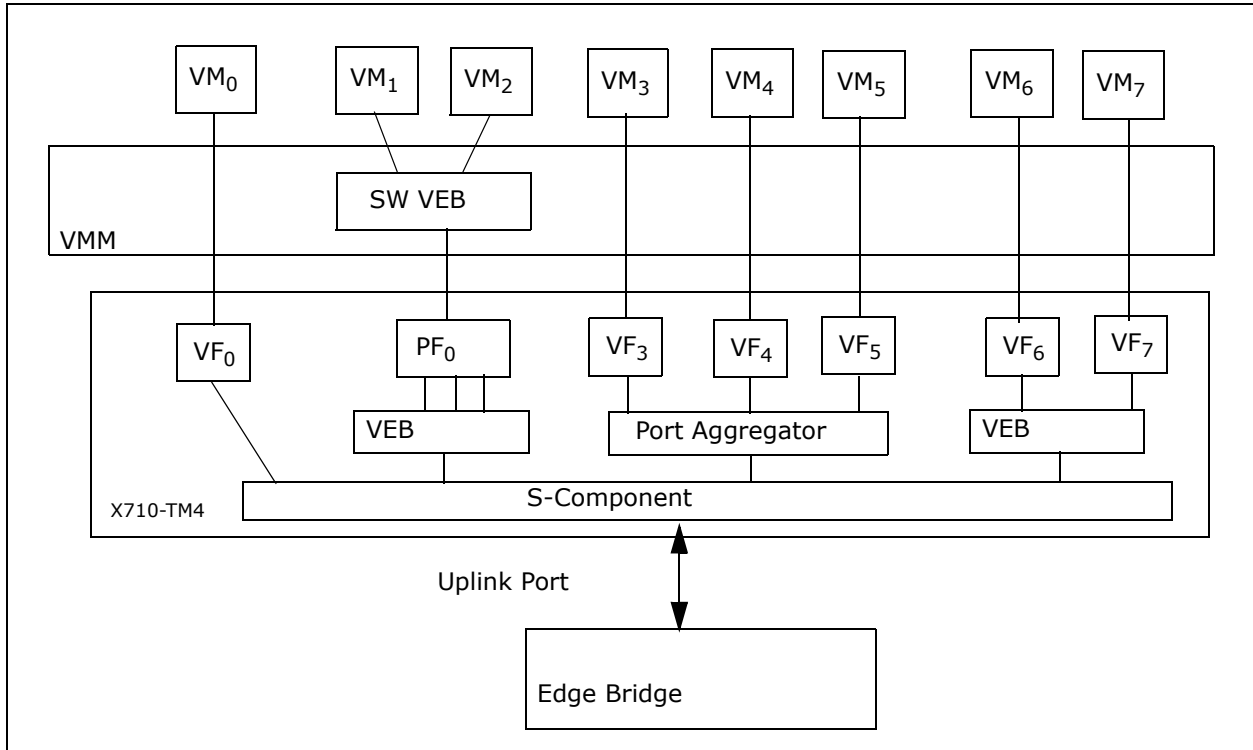


Figure 7-37. Cascaded Virtual Ethernet Bridge

7.4.2.4 Cascading of Software VEBs, Port Aggregators and Port Virtualizers

Software based VEBs, port aggregators and port virtualizers could be cascaded over hardware-based VEBs, PVs and PAs in the device. This section provides the typical cascaded configurations supported by the X710-TM4.

7.4.2.4.1 Cascaded Hardware and Software VEBs

Figure 7-38 shows a software VEB cascaded over a hardware VEB.

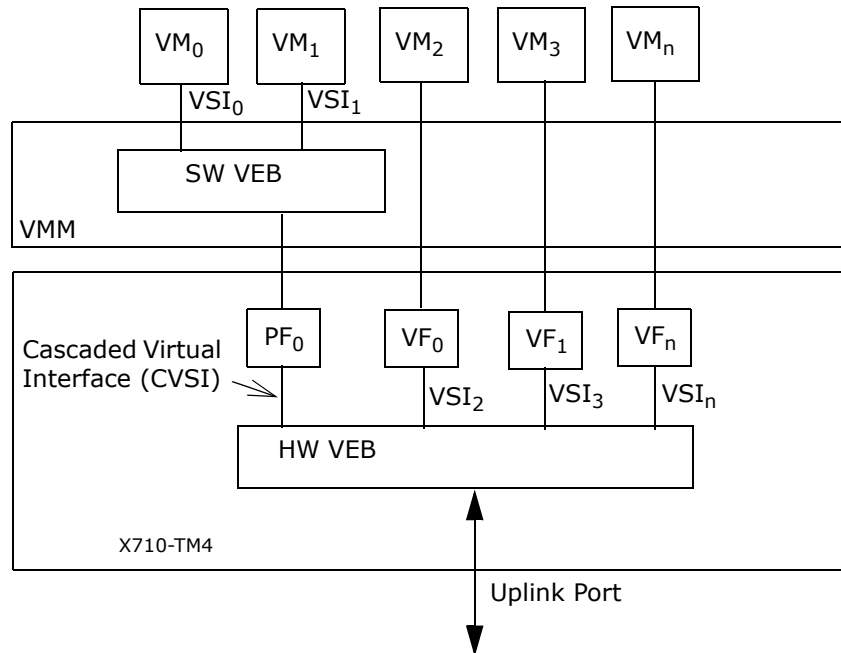


Figure 7-38. Cascaded Virtual Ethernet Bridge

In this configuration, a hardware VEB is instantiated in the X710-TM4 controller. The VEB is configured with multiple VSIs. A VSI refers to a connection between the VEB and a virtual end station and its associated resources such as MAC address, VLAN, BW/QoS attributes etc. Typically a VSI is configured with a single MAC address and VLAN. A software VEB is configured in the VMM and the connection between the software VEB and a port in the hardware VEB is referred to as cascaded virtual station interface (or simply a downlink port). The cascaded interface is a special VSI that is used as a downlink to a software VEB. This type of port is referred to as trunk port in typical Ethernet switches. The downlink ports are typically configured with multiple MAC addresses and multiple VLANs serviced by the software VEB. Security features such as MAC anti-spoofing and VLAN ingress checks (See [Section 7.4.6.1.1](#)) should not be configured on the downlink ports since those functions are performed at the ingress VSIs of a software VEB.

The X710-TM4 can be configured to perform VLAN offload functions on the downlink VSIs. In this case, the VLAN information is carried in the transmit descriptor and the X710-TM4 inserts the VLAN in the transmit direction. The X710-TM4 can strip the VLAN in the receive direction and store the VLAN tag information in the receive descriptor.

7.4.2.4.2 Cascaded Hardware and Software Port Aggregators

Figure 7-39 shows a software Virtual Ethernet Port Aggregator (VEPA) cascaded over a hardware VEPA.

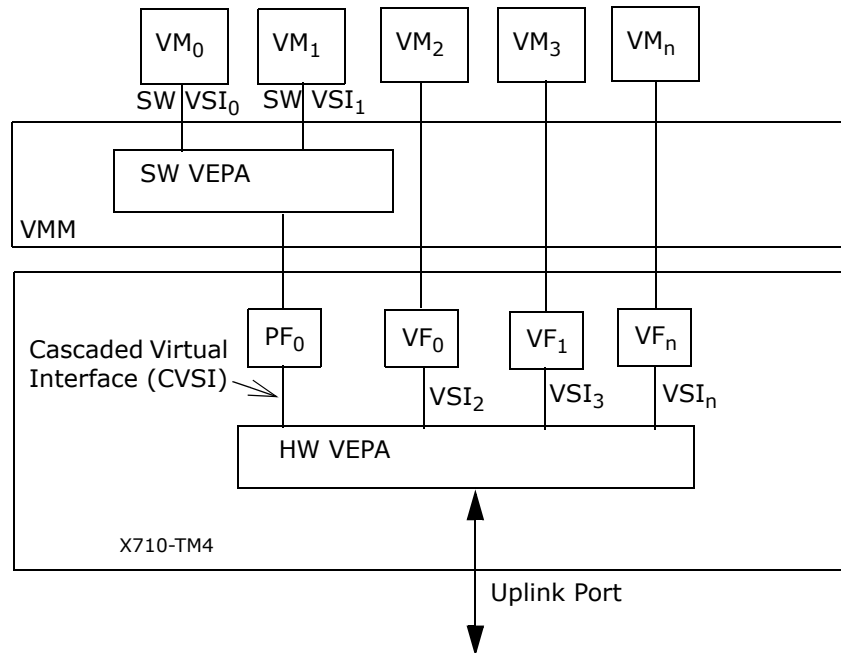


Figure 7-39. Cascaded VEPA

In this configuration a hardware VEPA is instantiated in the X710-TM4 controller. The hardware VEPA is configured with multiple VSIs. A software VEPA is configured in the VMM and the connection between the software VEPA and a port in the hardware VEPA is referred to as a cascaded virtual station interface (or simply a downlink port). The cascaded interface in this case is a special VSI that is used as a downlink to a software VEPA. The only cascaded configuration allowed over a hardware VEPA is a software VEPA. A software VEB cannot be cascaded over a hardware VEPA, this configuration is not allowed by the IEEE P802.3bg specification.

The downlink ports in a VEPA are similar to a VEB trunk port (see [Section 7.4.2.4.1](#)) except that multicast source address pruning functionality specified by IEEE P802.3bg is not configured on a downlink port to a software VEPA; this functionality is performed at the egress of the software VEPA VSIs connected to the VMs.

Security features such as MAC anti-spoofing and VLAN ingress checks (See [Section 7.4.6.1.1](#)) should not be configured on the downlink ports since those functions are also performed at the ingress VSIs of a software VEPA.

The X710-TM4 can be configured to perform VLAN offload functions on the downlink VSIs. In this case, the VLAN information is carried in the transmit descriptor and the X710-TM4 inserts the VLAN in the transmit direction. The X710-TM4 can strip the VLAN in the receive direction and optionally store the VLAN tag information in the receive descriptor.

7.4.2.4.3 Cascaded VEB and Port Virtualizers

Figure 7-40 shows a software VEB, hardware VEB and a software port virtualizer cascaded over a hardware port virtualizer. A port virtualizer logically extends the ports of an external bridge to virtual end stations as explained in Section 7.4.2.2.2. The X710-TM4 supports a hardware-based port virtualizer for directly connecting virtual end stations to extended bridge ports using VFs in SR-IOV configuration.

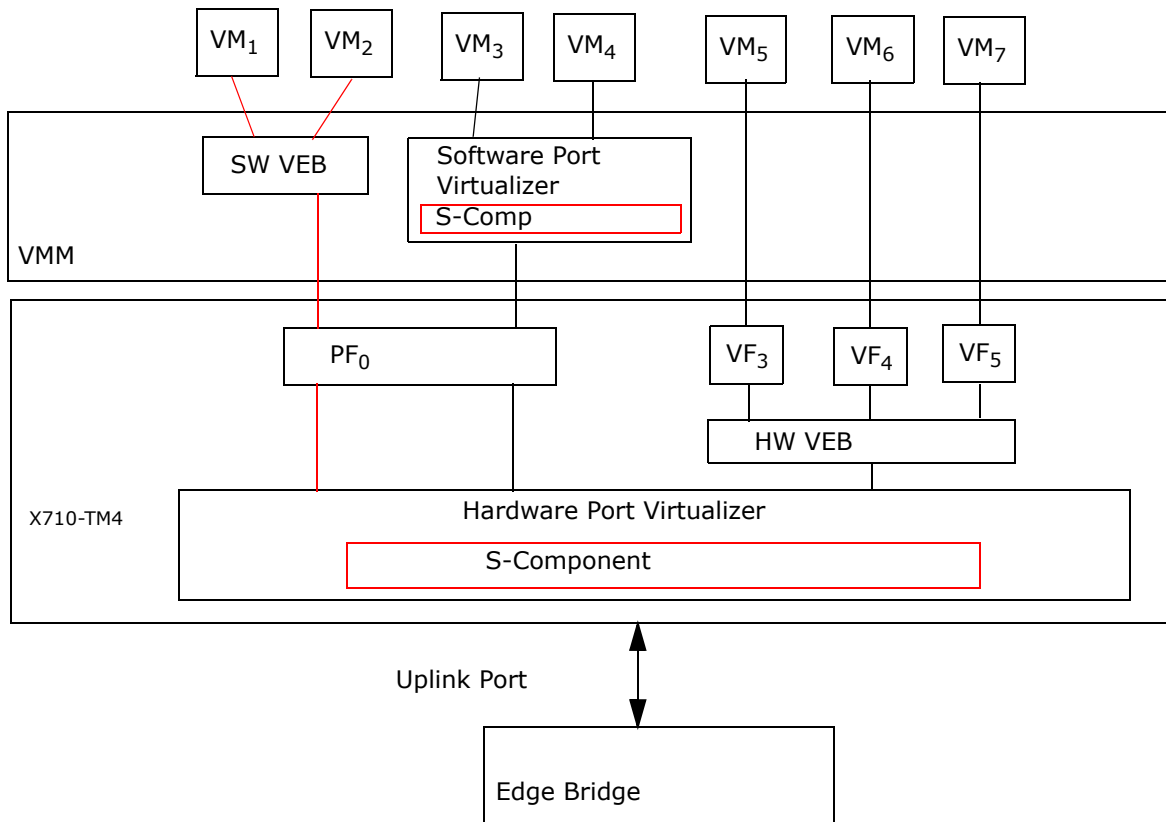


Figure 7-40. Cascaded VEB and Port Virtualizer



7.4.2.4.3.1 Cascaded Software Port Virtualizer Over Hardware Port Virtualizer

The X710-TM4 supports cascading a software-based port virtualizer in VMM to a hardware-based port virtualizer in the network controller. A software port virtualizer is configured in the VMM and the connection between the software port virtualizer and a port in the hardware port virtualizer is referred to as a cascaded virtual station interface (or simply a downlink port). The cascaded interface in this case is a special VSI that is used as a downlink to a software port virtualizer. A VSI connection between a VM and a port virtualizer uses single S-tag to identify the logical bridge port. However, a cascaded VSI (downlink port) connected to a software-based port virtualizer carries traffic to multiple software-based VSIs serviced by the software port virtualizer. So multiple S-tags need to be configured on this downlink VSI. Receive multicast replication functionality and source pruning functionality should be disabled on these downlink ports.

The X710-TM4 supports VLAN and S-tag offload functions for software-based port virtualizers connected to the downlink VSI (See [Section 7.4.5.2.1.2](#)). The X710-TM4 performs S-tag and VLAN tag insertion functionality on transmit direction and can optionally strip the VLAN tag and S-tags in the receive direction. In this case, the VLAN tag and S-tag information is carried in the transmit descriptor and the X710-TM4 inserts the VLAN tag and S-tag in the transmit direction. X710-TM4 can strip the VLAN tag and S-tag in the receive direction and store the VLAN and S-tag information in the receive descriptor.

The X710-TM4 can also classify ingress frames based on S-Tag and forward to selected queues within the downlink VSI connected to software port virtualizers (similar to VMDq1 mode). The X710-TM4 offload functions as previously described enhance the performance of software-based port virtualizers. See [Section 7.4.6.3](#) for additional details on software port virtualizer offload functionality.

Note: A VSI used as a cascaded port virtualizer should not activate L2 filters on the aggregated S-channels. The VSI should be defined with a connection type of default port.

7.4.2.4.3.2 Cascaded Software/Hardware VEB Over Hardware Port Extender (or Multichannel S-comp)

The X710-TM4 supports cascading of software VEBs to a hardware port virtualizer. In this configuration, a software VEB can be configured in the VMM and the connection between the software VEB and a port in the hardware port virtualizer is referred to as a cascaded virtual station interface (or simply downlink port). The X710-TM4 supports VLAN offload functions for software VEBs. The software VEB is not aware of S-tag functionality. The hardware port virtualizer (or multichannel S-component) performs the insertion and stripping of S-tag to the cascaded VSI (downlink port) connected to the VEB. The X710-TM4 can classify the ingress frames based on {VLAN, MAC address} pair and forward to selected queues within the downlink VSI connected to software VEBs (VMDq1 mode). See [Section 7.4.5.2.1.2](#) for additional details on offload functions available for software VEBs.

The X710-TM4 supports cascading of hardware VEBs to a hardware port virtualizer. A hardware VEB is configured in the X710-TM4 to support direct-connected VMs in SR-IOV mode. In this configuration, a hardware port virtualizer (or multichannel S-component) and a hardware VEB can be instantiated in the X710-TM4. The hardware VEB can be connected through a cascaded (or downlink) port in the hardware port virtualizer. Logically the hardware VEB's trunk port is connected to the external controlling bridge through a logically extended bridge port identified by an S-tag. The hardware port virtualizer (or multichannel S-component) performs insertion and stripping of S-tags to the cascaded downlink port connected to the hardware VEB.



7.4.3 Edge Virtual Bridging (EVB) Management Protocol

This section describes the control and management protocol for discovery and configuration of EVB components between the end stations and external edge switches. The current IEEE 802.1Qbg EVB supports: VEBs and Virtual Ethernet Port Aggregators (VEPAs) for virtual networking. A control protocol is required to discover the capabilities of the end stations and external switches and configure the appropriate switching elements at both ends.

To enable EVB in the device, all of the following must be set:

- The `GLGEN_STAT.EVBEN` bit must be set to 1b either from the NVM or through soft SKU.
- The NVM *EVB Protocols Enabled* field should be set to 1b.
- The NVM *Switching Mode* field should be set to the *EVB Switching* value.

EVB is configured and manages via the following stages:

- S-channels are enabled, defined and enumerated between the endpoint (station) and switch (bridge) by exchanging CDCP TLVs over LLDP. Device sets a default VSI per S-channel.
- Software might then instantiate Edge Relay (ER) bridges, one per each S-channel.
- Support for the relay protocol (VEB/VEPA) of each ER is negotiated through LLDP EVB TLVs. Reflective relay might be enabled and common variables for VDP/ECP are negotiated.
- A reliable control protocol [Edge Control Protocol (ECP)] is then established.
- A VSI discovery and configuration protocol (VDP) runs over ECP to associate VSIs with bridge ports.

7.4.3.1 S-tag Allocation Through CDCP TLVs

CDCP TLVs are exchanges over LLDP. The structure of the LLDP packet and the CDCP TLV are described in [Section 7.12](#).

Following initialization of an LLDP agent, and once the conditions for CDCP support are met, a local copy of the CDCP TLV is established and exchanged with the other end. Once the device receives a S-tag setting from the remote CDCP TLVs, it continues with EVB configuration. Specifically, the S-tags extracted from the CDCP MIB are handled as if received through CLP strings or NCSI depends on customer.

Mapping of the SCID/SVID pairs might change as a result of the following events:

- The LLDP agent reverts to the local MIB event as described in [Section 7.12.4.3](#). In such a case, all SCID/SVID pairs are deleted.
- A CDCP TLV is received with a modified list of SCID/SVID pairs. SCID/SVID pairs might be removed, modified, or added.

7.4.3.2 VSI Setup Through VPD TLVs

A new protocol, VSI Discovery Protocol (VDP), is defined by IEEE. The VDP protocol is transported over Edge Control Protocol (ECP) that enables an exchange of VSI-specific TLVs between the edge switch and the control plane of an appropriate switching element. The TLV transfer protocol is expected to use the same group multicast destination address(es) reserved for LLDP as previously described with a new Ethertype. A new Ethertype enables forwarding the protocol frames to the appropriate control plane of the respective switching element.



The VDP protocol uses TLVs to discover and configure the VSIs in the end station. The VDP protocol is used for establishing a new VSI and exchanging VSI attributes such as VSI ID, MAC, VLAN, QoS parameters, etc. The VDP protocol might also be used for exchanging VSI statistics from end station to edge switches. The X710-TM4 has the ability to filter for appropriate EVB control frames and forward to appropriate management ports (VSIs or VSI queues) of the respective switching elements. The control frames (such as LLDP or VDP) are expected to be exchanged by the respective control plane agent in the VMM (or IOVM) and the external switch. LLDP is a link-level protocol, so a VM is not expected to directly exchange LLDP control frames with an external switch. For example, the control frames cannot cross the virtual bridge. If a VM is configured to exchange control frames, it does so with the appropriate virtual bridge control agent in the VMM. The X710-TM4 has a mechanism to either filter and/or forward the control frames from the VSIs to the management port of the switching element to which the VM is connected.

When a VEB or VEPA is instantiated over a multi-channel S-component, this is the equivalent of a VEB or VEPA uplink directly connected to a logical port in the external bridge so the external bridge can exchange bridge control frames over the logical link to the VEB or VEPA. In this case, the external bridge can send or receive LLDP frames to a VEB or VEPA in the end station. These frames are identified using the S-tags that correspond to the extended bridge ports. Any of the bridge control frames can pass through this S-channel (like LACP, new TLV transfer protocol/VDP, etc.). The bridge control frames over the S-channels are not expected to be VLAN tagged with an inner VLAN, so these control packets are untagged when delivered to the VEB or VEPA by the S-component. The X710-TM4 can forward those control frames to the respective VEB or VEPA control ports (VSIs).

So in a multi-channel configuration there could be multiple LLDP agents (peer-to-peer) established between the external bridge through extended bridge ports (S-channels) and the corresponding VEB, VEPA or port virtualizer components in the end station.

Note: After a link down/link up event, the switch configuration protocol should be run again, as the partner (and its capabilities) might have changed.

7.4.4 Switch Elements

Figure 7-41 shows the possible switching elements on one of the X710-TM4 ports and the connections between them. Every switching element is optional. In a minimal case, the Ethernet port might be connected to a PCI function (VSI) with no switching capabilities at all. In this minimal case, an L2 filter is used as in regular devices.

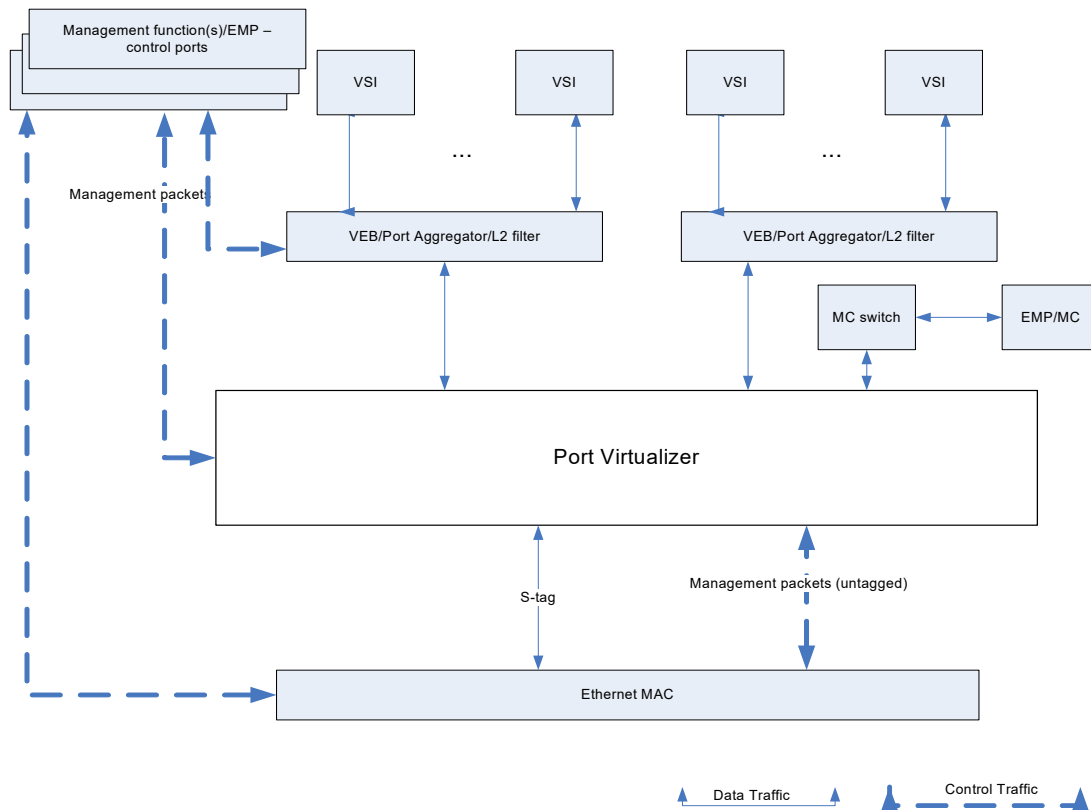


Figure 7-41. Switching Elements Per Port

There can be up to one port virtualizer per physical port. There can be up to 16 concurrent VEB or port multicasting defined in the device. There can be up to one L2 filter per VSI.

7.4.4.1 MAC

The functionality of the MAC (CRC handling, error detection, etc.) is described in [Section 3.2.1](#). This section only describes the switching decisions of the MAC.

The MAC switching behavior is set according to the following parameters:

- Port enable
- Link status
- Save bad packets (stored in the *PRT_SBPVSI.SBP* field)\
- Default port VSI (stored in the *PRT_SWT_DEFPORPTS.DEFAULT_VSI* field)
- Local MAC address (per LAN port)



- One or more control ports of the MAC. If a port virtualizer or a VEB is directly connected to this MAC, they might share the same control ports. At initialization time, a single control port is connected to the EMP. At a later time, if requested by the software device driver, an additional control port on a VSI of one of the PFs might be added or might replace the EMP control port.

As a rule, the MAC forwards all the packets to the next stage, apart from the following packets:

- Packets with L2 error — these packets are dropped, unless the *Save Bad Packets* attribute of the MAC is set. In this case, the MAC must define a control port and all the error packets are forwarded to this port. The errors included in this category are:
 - CRC errors
 - Alignment errors
 - Length errors (either too big or too small)
- Packets with the MAC local MAC address. If a *Station MAC* address is defined, packets with this address might be forwarded to the one of the control ports of the MAC in combination with some Ethertype filters. Each control port might require forwarding of these packets using the *Add Control Packet Filter* or the *Add MAC, VLAN pair* admin commands.
- Link local packets without S-tag:
 - Flow control packets (both 802.3x link flow control and 802.3bd priority flow control frames) are handled by the MAC and are not forwarded.
 - Any packet defined in a control VSI forwarding rules (see [Section 7.4.5.2.2.2](#)).

7.4.4.2 Port Virtualizer

An Ethernet port can be connected to a port virtualizer.

A port virtualizer can be either an S-comp ([Section 7.4.4.2.1](#)) or a port extender ([Section 7.4.5](#)). Each port virtualizer should be connected to a control port that receives the control protocol (CDCP) packets.

There can be one port virtualizer per physical port. These two elements are mutually exclusive and in a given device, only one type of port virtualizer can be initiated on all ports.

A port virtualizer can be created using the *Add Port Virtualizer* admin command described in [Section 7.4.9.5.6.1](#).

Note: If a port virtualizer is not implemented on a port, S-tagged packets are dropped.

7.4.4.2.1 Service-VLAN Component (S-comp)

An S-component is basically a mux/demux component that divide the Ethernet port to a set of S-channels.

Each channel in the S-component can be attached as a physical port to one of the switches. If no switch is needed, an S-comp is used to directly connect S-channels to VSIs.

7.4.4.2.1.1 S-tag Format

The S-tag format is the format defined for S-tags in the IEEE 802.1ad specification as follows:



Table 7-44. S-tag Format

0	1 5	1 6	1 8	19	2 0	3 1
0x88-a8/0x81-00	PCP		DE I		S-VID	

- 0x88-A8 or 0x81-00— S-tag Ethertype. The Ethertype used is defined according to the *Channel Identifier* field in the features enable word in the EMP SR settings module header (0x0 = 0x88A8, 0x1=0x8100).
- PCP— Priority Code Point. The priority bits of the S-tag. These bit can be derived from the VLAN priority bits as described in [Section 7.4.6.2](#).
- Drop Eligible Indicator (DEI) —This bit is ignored.
- S-VID — the S-tag ID.

7.4.4.2.1.2 S-comp Receive Flow

S-comp acts only on packets with an S-tag received from the uplink. These packets are forwarded to one of multiple S-comp channels according to the S-tag forwarding table, thus forwarding them directly to VEB switches or VSIs.

If no VEB is defined, then the VSI connected to the S-channel should be put in promiscuous mode using the Set VSI Promiscuous Modes command in order to receive all the traffic with the given S-tag. If an S-component is defined in the system, packets without S-tag or with an unrecognized S-tag not forwarded to one of the control ports are sent to the default port. If such a port is not defined, they are dropped.

The S-tag is usually removed by the S-component and is not forwarded to the host or to the sideband interface. This functionality is controlled by the SS-tag extract mode parameter of the Add VSI admin command. For cascaded S-comp ports that represents multiple S-channels, the S-tag can be either left in the packet or removed and stored in the *L2TAG2 (1st)* field of the receive descriptor. The *L2TAG2 (1st)* field of the receive descriptor contains the S-tag if the S-tag extract mode parameter of the VSI is set to 10b and is valid if the *L2TAG2P* flag in the descriptor is set. If S-tag extraction to descriptor is required the VSI must use 32 bytes receive descriptors.

The algorithm used by an S-comp is described in [Section 7.4.8.2](#).

7.4.4.2.1.3 S-comp Transmit Flow

The S-comp adds the S-tag matching to the S-channel from which the transmit packet is received and forwards it to the physical port. It also forwards packets received from the control ports to the physical port.

Switch-based insertion of the S-tag can be disabled using the S-tag insert enable parameter of the Add VSI admin command. For cascaded S-comp ports that represents multiple S-channels. the S-tag insertion should be disabled. In this case, software might either send an untagged packet, request S-tag insertion via the transmit descriptor or add it inline in the packet. This functionality is enabled by the Accept Tag from host parameter in the Add VSI admin command.

An S-comp only transfers packets from the S-channels to an Ethernet link or vice versa. It does not replicate packets or forward packets between virtual ports or physical ports. A VEB connected on top of an S-channel can forward packets between VSIs contained by this VEB.



7.4.4.2.2 Port Virtualizer (S-comp) Parameters

A port virtualizer is defined by the following parameters:

1. Port virtualizer connectivity:
 - a. Uplink connected.
 - b. Control ports that receive link local packets. The control ports might be VSIs or the internal embedded controller or both.
2. Demux rule — what is the rule used to demux ingress packets. The usual rule is to use an S-tag as previously described.
3. Default port— This port receives all the traffic not identified as control packets and not forwarded to any of the S-channels. If this port is not defined, such traffic is dropped.
4. Set of channels —Each channel is defined by the following parameters:
 - a. The connected VSI or next level switch.
 - b. An S-tag (S-VID) - see [Section 7.4.4.2.1.1](#).
 - c. A maximal and a minimal rate.
 - d. S-tag strip enable.
 - e. S-tag insert enable.
 - f. Report S-tag.
 - g. Accept tag from host.

7.4.4.2.3 LAN/SAN S-comp

The LAN/SAN port extender divides the uplink port between LAN and SAN traffic. The LAN channel might be attached to a switch (like a VEB).

This S-comp differs from the standard 802.1Qbg port extender in that it uses the MAC address as the channel classifier. The MAC address behaves the same way as the S-tag, and it can translate to a single switch ID. In this case, the traffic is untagged (no S-tag).

7.4.4.2.3.1 Receive Flow

The S-comp filters on unicast and multicast addresses associated with the SAN function. A packet is forwarded to the SAN channel if it matches one of the MAC addresses (unicast or multicast) associated with SAN. Else, it is forwarded to the LAN channel.

Note: The MAC address is used twice in the switch process, first to define the S-channel and second to define the VSI within this S-channel.

Contrary to other port extenders, the MAC address is never stripped and it is always forwarded to the host.

The algorithm used by a LAN/SAN S-comp is described in [Section 7.4.8.1](#).

7.4.4.2.3.2 Transmit Flow

The S-comp only transfers packets from the S-channels to an Ethernet link or vice versa. It does not replicate packets or forward packets between virtual ports or physical ports. Also, no tag is added to the transmitted packet.



7.4.4.2.4 Port Virtualizer Control Ports

The port virtualizer might forward control packets to the associated control port.

Port virtualizer control packets are untagged LLDP or enhanced LLDP packets combined with specific MAC addresses as described in [Section 7.4.3](#). Each control port might require forwarding of specific control packets forwarding using the Add Control Packet Filter admin command.

In MFP mode, this control port is owned by the EMP. In SFP mode, it might be owned by the PF driver or by internal firmware.

7.4.4.2.5 Port Virtualizer Flow Diagram

[Figure 7-42](#) shows the high-level flow used to define the switch ID associated with a packet. Details can be found in the algorithm sections ([Section 7.4.8](#)).

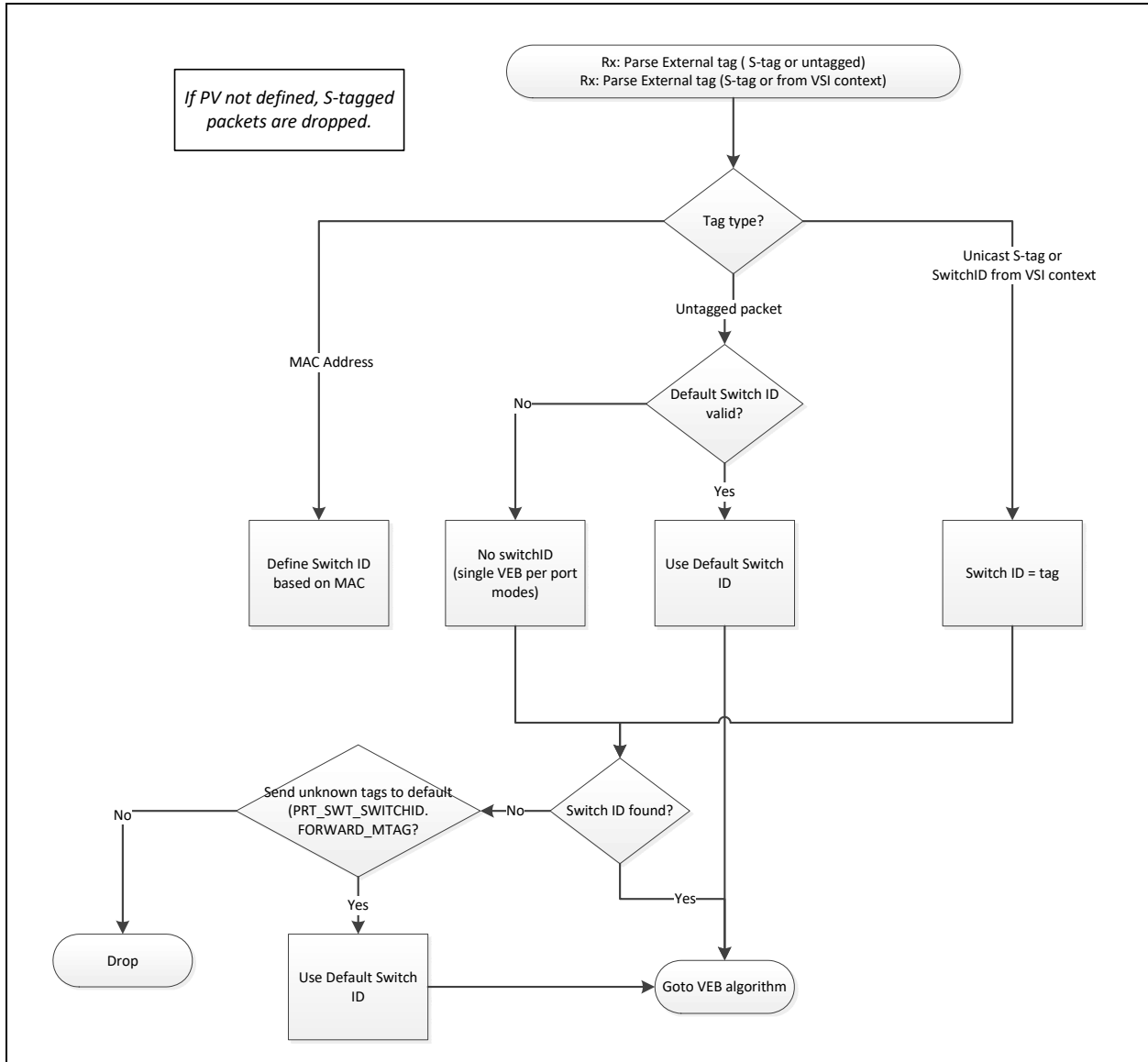
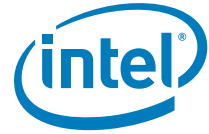


Figure 7-42. Port Virtualizer Flow

7.4.4.3 L2 Filters

A VSI directly connected to a physical port, to a port extender or to an S-comp, still needs the ability to filter incoming traffic as done in traditional devices. The L2 filtering element provides this ability. An L2 filtering element is relevant only for receive or sideband traffic and does not apply to transmit traffic. An L2 element has no control port and is controlled directly by the function.

An L2 filter element is not represented as part of the topology and is implicitly created by adding L2 filters to existing VSIs.



This section describes L2 filters used when no VEB or VEPA element is initiated. If a VEB or VEPA element is added, the L2 filtering is done as part of the VEB forwarding as described in the following sections.

An L2 filtering element includes a set of filters used to define whether a packet is received by the function or is dropped. There are two type of filters: exclusive filters that identify the packet as exclusive to this function and non-exclusive filters that might pass also packets that might be forwarded to other ports or functions.

Non-exclusive traffic might be further defined as perfect or imperfect, where perfect means that a packet that passed these filters was requested by the host and imperfect filters might pass packets the host did not request.

In addition, a packet might be filtered out if it is larger than the maximum size defined by the port.

Note: Even if a packet passed the length filter of the port, it might yet be filtered by the length limitation of the receive queues.

An L2 filter is applicable only to receive traffic and does not refer to transmit traffic. Thus, if there is only an L2 filtering element, transmit traffic is forwarded to the network. A packet sent by this connection is looped back to the sending VSI.

Note: The cloud filtering includes an L2 filter as part of its VEB or VEPA. This L2 filter is described in [Section 7.4.4.3](#).

7.4.4.3.1 Exclusive Filters

The following exclusive filters are available as part of an L2 filtering element:

- {MAC:VLAN} Unicast table— Used to forward to a port packets matching both the MAC and VLAN pair. A VLAN value of zero includes untagged packets. Only unicast addresses are considered as exclusive.
- {MAC} Unicast table— Used to forward to port packets matching MAC addresses ignoring the VLAN tag.
- {VLAN} table— Used to define the VLAN to which the port belongs. A VLAN value of zero indicates the port can receive untagged packets.
- {Ethertype} table— Used to forward to port packets matching an Ethertype ignoring the MAC address and VLAN tag.

7.4.4.3.2 Non-exclusive Filters

The following non-exclusive perfect filters are available as part of an L2 filtering element:

- {MAC:VLAN} Multicast table — Used to forward to port packets matching both the MAC and VLAN pair. A VLAN value of zero includes untagged packets. Only unicast addresses are considered as exclusive.
- {MAC} Multicast table — Used to forward to port packets matching MAC addresses ignoring the VLAN tag.
- {UPE}— Promiscuous unicast.
- {MPE}— Promiscuous multicast.
- {BAM}— Accept broadcast packets.

The algorithm used by an L2 filter is described in [Section 7.4.8.3](#).



7.4.4.4 VEB

A VEB is used to switch packets between a set of virtual ports, a physical port or an S-channel and a control port. The VEB behaves like a managed 802.1d transparent switch. All the configuration of the switch is done by a software agent. There are no auto configuration capabilities in the VEB. For example, there is no auto-learning of MAC addresses.

A VEB can forward packets received from any of its ports to a list of ports (including, optionally the originating port). The VEB uses the rules described in [Section 7.4.4.4.2](#) to define the egress ports of each packet.

A packet is associated with a VEB using the switch ID associated to it. A switch ID can be either (0,S-tag)} or a switch ID based on the SwitchID VSI parameter or the port SWID parameter (PRT_SWT_SWITCHID.SWID).

There can be up to 16 VEBs or port aggregators in the X710-TM4. They can be assigned dynamically to PFs. All the virtual ports associated with a VEB must belong to a single PF or its VFs.

Note: A VEB can be added only by a PF and can not be added by a VF or the EMP.

A VEB can be created using the Add VEB admin command described in [Section 7.4.9.5.7.1](#).

7.4.4.4.1 VEB Parameters

A VEB is defined by the following parameters:

1. A control port that receives link local packets. For example, 802.1X packets or LLDP packets.
2. An optional default port that might receive packets received from the physical port and not forwarded to any virtual port.
3. Definitions of local and private VLANs.
 - a. Local VLANs: A list of VLAN tags defined as local that do not include the physical port.
 - b. Private VLANs: A list of VLAN tags defined as private. For each private VLAN tag, define the promiscuous ports list, the community ports list, and the isolated ports list.

Note: More details on special VLANs can be found in [Section 7.4.6.1.2](#).

4. A set of VSIs - Each VSI is defined by the following parameters:
 - a. The connected PCIe function.
 - b. A transmit enable and receive enable.
 - c. A set of forwarding tables and parameters rules used to forward packets to the port as described in [Section 7.4.4.4.2.1](#).
 - d. Security features
 - Port based VLAN insertion. See [Section 7.4.6.1.2](#) for more details.
 - Anti spoofing enables. See [Section 7.4.6.1.1](#) for more details.
 - e. Optionally, a maximum and a minimum rate.

The full list of VSI parameters is described in [Section 7.4.5.2.5](#).

7.4.4.4.2 VEB Switching Rules

The following parameters are used to define which egress ports (VSIs and LAN) receive a packet received by the VEB.



7.4.4.4.2.1 Forwarding Tables and Parameters

- Priority 1 filters (Control filters):
 - {Ethertype} table: Used to forward to port(s) packets matching an Ethertype ignoring the MAC address (usually used for the control VSI).
 - {MAC, Ethertype} table: Used to forward to port(s) packets whose MAC address match a given MAC address and Ethertype match an Ethertype (usually used for the control VSI).
- Priority 2 filters (ATQ filters, if enabled):
 - L4 port
 - {Destination IP, L4 port}
 - {Outer MAC, L4 port}
 - {Outer MAC, Outer VLAN, L4 port}
 - {Inner MAC, L4 port}
 - {Inner MAC, Inner VLAN, L4 port}
- Priority 3 filters (Cloud filters):
 - {Inner MAC, Inner VLAN} (for NVGRE, VXLAN or Geneve packets)
 - {Inner MAC, Inner VLAN, Tenant ID} (for NVGRE, VXLAN or Geneve packets)
 - {Inner MAC, Tenant ID} (NVGRE packet or VXLAN Geneve packets).
 - Inner MAC filter
 - {Outer MAC, Tenant ID, Inner MAC} filter
 - Inner IP filter
 - {Inner Source IP, inner destination MAC} filter.

Note: The network key is extracted from the GRE or UDP headers in MAC in GRE or MAC in UDP.

GRE key is a 4-byte field enabled by the 'k' flag in the GRE header. The key is extracted only if the GRE header includes a key as indicated by active 'k' flag in the header.

The UDP port used for MAC in UDP tunneling is defined by a shared 16-entry table with tunneling port numbers. The key is extracted if the UDP port number contains a key as defined by the UDP protocol index, which is programmed by the Add Tunneling Port command.

- Priority 4 filters (L2 filters):
 - {MAC, VLAN} table: Used to forward to VSI(s) packets matching both the MAC and VLAN pair.
 - {MAC} table: Used to forward to a VSI(s) packets matching MAC addresses ignoring the VLAN tag.
 - {VLAN} table: Used to define the VLANs to which a VSI(s) belongs for egress and ingress checks. This table can point to two lists - one list that is used for ingress VLAN filtering and one used for egress VLAN filtering.

Note: In all previous tables, a VLAN value of zero includes untagged packets.

Note: A port in these tables means either a VSI or the LAN port for transmit packets and a VSI for receive packets.

- Promiscuous modes per VSI:
 - {UPE}: Used to forward to a VSI, all unicast packets.
 - {MPE}: Used to forward to a VSI, all multicast packets.
 - {BAM}: Used to forward to a VSI, broadcast packets.
- {VPE}: Used to forward to a VSI, packets with any VLAN tag.
- Loopback rules:



- {ALLOWLOOPBACK} Should this port be allowed to send packets to other virtual ports?
- {PruneEnable} Should this port receive packets it sent? This mode is useful to enable an offload of a software switch connected to this virtual port.
- Anti spoofing parameters. These parameters defines if a packet should be allowed to enter the switch.
 - MAC Anti spoofing enable
 - VLAN anti spoofing enable

The forwarding filters are divided to groups. Those filters that are needed for tunneled packet formats and those ones that are shared for all packet formats (based on the outer MAC,VLAN). Filters that match tunneled packet formats take precedence on those filters that matches all packet formats (as shown in [Figure 7-45](#)).

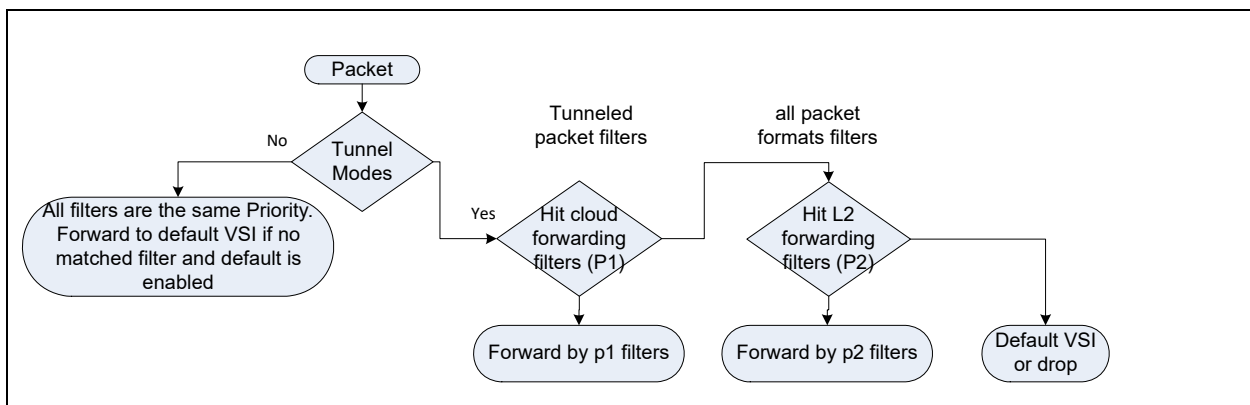


Figure 7-43. Cloud Forwarding Filters Priority

The Add Control Packet Filter ([Section 7.4.9.5.9.9](#)) and Remove Control Packet Filter ([Section 7.4.9.5.9.10](#)) commands are used to add or remove control filters settings. Each control filter can point to a single VSI. There is no support for packet replication based on control filters. An error is returned if an existing control filter is added to point to another VSI.

The Add Cloud filters ([Section 7.4.9.5.9.11](#)) and Delete Cloud filters ([Section 7.4.9.5.9.12](#)) commands are used to control cloud specific filters settings. Each cloud filter can point to a single VSI. There is no support for packet replication based on cloud filters. In addition to those filters, in order for a packet to be received by one of the VSIs, it must also pass the {L2 MAC} filter. In order to enable L2 filtering, the Flags.Enable L2 filtering bit in the Add VEB command should be set when creating the cloud VEB.

Note: This L2 filtering is not related to the L2 filtering defined in [Section 7.4.4.3](#).

- The Add MAC, VLAN pair ([Section 7.4.9.5.9.1](#)), Remove MAC, VLAN pair ([Section 7.4.9.5.9.2](#)), Add VLAN ([Section 7.4.9.5.9.3](#)), and Remove VLAN ([Section 7.4.9.5.9.4](#)) commands are used to set regular MAC/VLAN filters.

Note: The Add MAC, VLAN pair command allows filtering based on a MAC address (any VLAN) or on a MAC, VLAN pair. The Add VLAN command is used to define the VLAN membership of ports and is not used to add destination VSIs. Using this method, the VEB can allow MAC based filtering and promiscuous modes within specific VLANs.

- The Set VSI promiscuous modes ([Section 7.4.9.5.9.5](#)) command is used to set promiscuous filters.
- The Add Mirror Rule ([Section 7.4.9.5.10.1](#)) and Delete Mirror Rule ([Section 7.4.9.5.10.2](#)) commands are used to control mirror rules.



Note: It is assumed that manageability packets and control port packets will not be encapsulated.

7.4.4.4.3 Adding L4 Port Filters

The internal switch configuration can be changed to define L4 port filters in place of existing cloud filters by using the Set Switch Configuration command. This change is global for the device and cannot be reversed except by a core reset. An L4 port filter is added using the big buffer option of the Add Cloud Filters command, and the L4 port value is placed in bytes 110-111 of the command buffer.

The following modes are supported:

- Mode 0: No L4 port filters; all cloud filters are available. This is the default mode after core reset.
- Mode 1:
 - L4 port filters
 - L4 port
 - Cloud filters removed: None
- Mode 2:
 - L4 port filters
 - {Outer MAC, L4 port}
 - {Outer MAC, Outer VLAN, L4 port}
 - {Destination IP, L4 port}
 - Cloud filters removed
 - Inner MAC
 - {Outer MAC, Tenant ID, Inner MAC}
- Mode 3:
 - L4 port filters
 - {Inner MAC, L4 port}
 - {Inner MAC, Inner VLAN, L4 port}
 - {Destination IP, L4 port}
 - Cloud filters removed
 - Inner MAC
 - {Outer MAC, Tenant ID, Inner MAC}

Note: For modes 1-3, the L4 port can be specified as the source port or the destination port in the Set Switch Configuration command.

Note: For tunneled packets in modes 1-3, the destination IP and L4 port values are taken from the inner L3/L4 headers.

7.4.4.4.4 VEB Flow

Figure 7-46 shows the generic flow of the VEB element. The exact algorithm implementing this flow is described in Section 7.4.8.4.

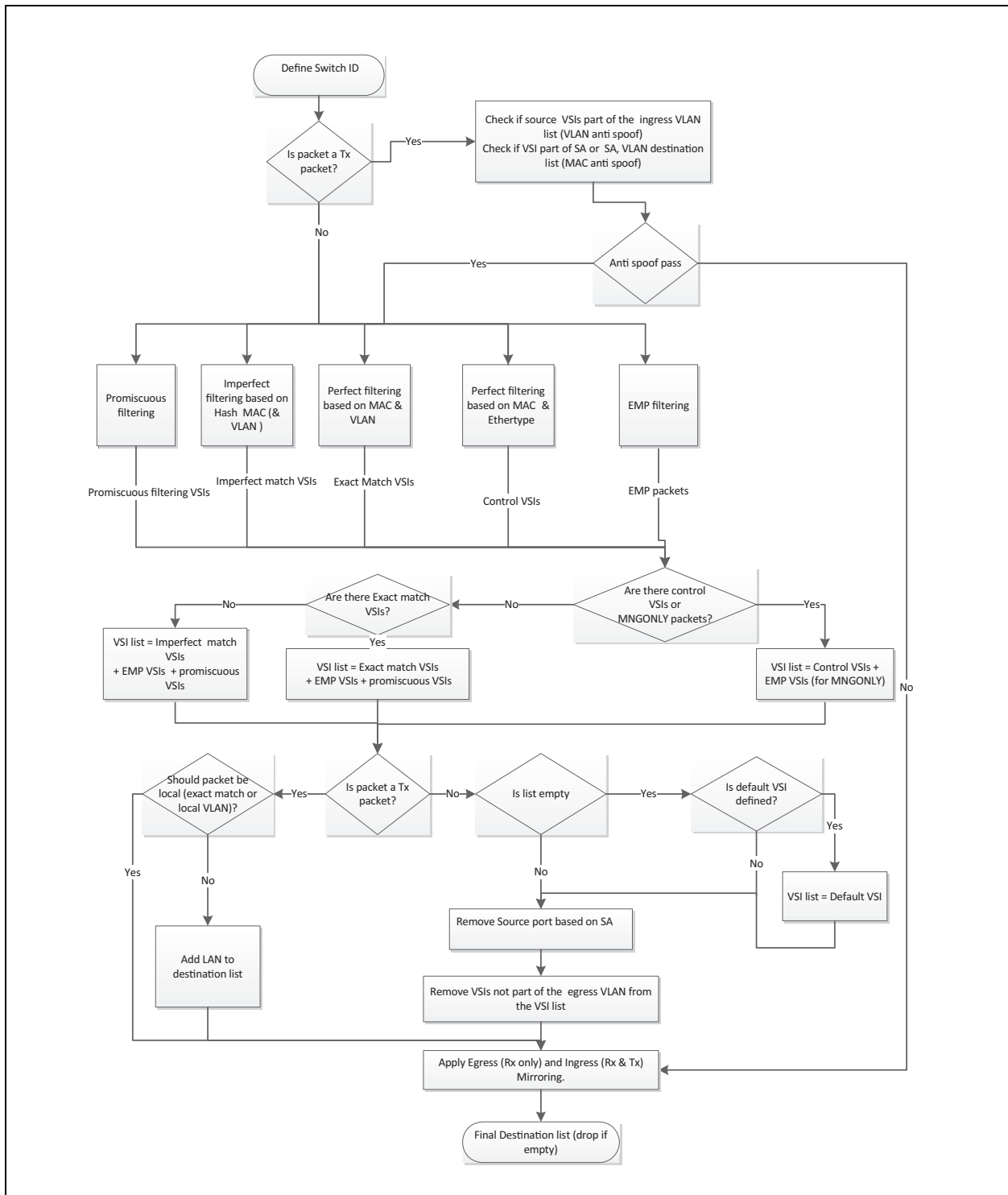


Figure 7-44. VEB Flow Diagram



7.4.4.5 VEB with Cloud Support (Cloud VEB)

A VEB is used to switch packets between a set of virtual ports, a physical port or an S-channel and a control port. The VEB behaves like a managed 802.1d transparent switch. All the configuration of the switch is done by a software agent. There are no auto-configuration capabilities in the VEB. For example, there is no auto-learning of MAC addresses.

A VEB can forward packets received from any of its ports to a list of ports including, optionally, the originating port. The VEB uses the rules described in [Section 7.4.4.5.2](#) to define the egress ports of each packet.

A packet is associated with a VEB using the switch ID associated to it. A switch ID can be either {0,S-tag} or a switch ID based on the SwitchID VSI parameter or the port SWID parameter (PRT_SWT_SWITCHID.SWID).

There can be up to 16 VEBs or port aggregators in the X710-TM4. They can be assigned dynamically to PFs. All the virtual ports associated with a VEB must belong to a single PF or its VFs.

Note: A VEB can be added only by a PF and cannot be added by a VF or the EMP.

A VEB can be created using the Add VEB admin command described in [Section 7.4.9.5.7.1](#).

A cloud VEB can also be configured to VEPA mode. The exact algorithm used by a cloud VEB is described in [Section 7.4.8.4](#).

7.4.4.5.1 VEB with Cloud Support Parameters

A VEB is defined by the following parameters:

1. A control port that receives link local packets (such as 802.1X packets or LLDP packets).
2. An optional default port that might receive packets received from the physical port and not forwarded to any virtual port.
3. Definitions of local and private VLANs.
 - a. Local VLANs— A list of VLAN tags defined as local that do not include the physical port.
 - b. Private VLANs— A list of VLAN tags defined as private. For each private VLAN tag, define the promiscuous ports list, the community ports list and the isolated ports list.

Note: More detail on special VLANs can be found in [Section 7.4.6.1.2](#).

4. A set of VSIs — Each VSI is defined by the following parameters:
 - a. The connected PCIe function.
 - b. A transmit enable and receive enable.
 - c. A set of forwarding tables and parameters rules used to forward packets to the port as described in [Section 7.4.4.5.2](#).
 - d. Security features.
 - Port-based VLAN insertion (see [Section 7.2.4.4](#) for more detail).
 - Anti spoofing enables (see [Section 7.4.6.1.1](#) for more detail).
 - e. Optionally, a maximum and a minimum rate.

The full list of VSI parameters is described in [Section 7.4.5.2.5](#).



7.4.4.5.2 Cloud VEB Switching Rules

The following parameters are used to define which egress ports (VSIs and LAN) receive a packet received by the VEB.

7.4.4.5.2.1 Forwarding Tables and Parameters

- Priority 1 filters (Control filters):
 - {Ethertype} table—Used to forward to port(s) packets matching an Ethertype ignoring the MAC address (usually used for the control VSI).
 - {MAC, Ethertype} table —Used to forward to port(s) packets whose MAC address match a given MAC address and Ethertype match an Ethertype (usually used for the control VSI).
- Priority 2 filters (ATQ filters, if enabled):
 - L4 port
 - {Destination IP, L4 port}
 - {Outer MAC, L4 port}
 - {Outer MAC, Outer VLAN, L4 port}
 - {Inner MAC, L4 port}
 - {Inner MAC, Inner VLAN, L4 port}
- Priority3 filters (cloud filters):
 - {Inner MAC, Inner VLAN} (for NVGRE, VXLAN or Geneve packets)
 - {Inner MAC, Inner VLAN, Tenant ID} (for NVGRE, VXLAN or Geneve packets)
 - {Inner MAC, Tenant ID} (NVGRE packet or VXLAN/Geneve packets).
 - Inner MAC filter
 - {Outer MAC, Tenant ID, Inner MAC} filter
 - Inner IP filter
 - {Inner Source IP, inner destination MAC} filter.

Note: The network key is extracted from the GRE or UDP headers in MAC in GRE or MAC in UDP packets.

GRE key is a 4-byte field enabled by the 'k' flag in the GRE header. The key is extracted only if the GRE header includes a key as indicated by active 'K' flag in the header.

The UDP port used for MAC in UDP tunneling is defined by a shared 16-entry table with tunneling port numbers. The key is extracted if the UDP port number contains a key as defined by the UDP protocol index, which is programmed by the Add Tunneling Port command.

- Priority4 filters (L2 filters):
 - {MAC, VLAN} table— Used to forward to VSI(s) packets matching both the MAC and VLAN pair.
 - {MAC} table— Used to forward to a VSI(s) packets matching MAC addresses ignoring the VLAN tag.
 - {VLAN} table— Used to define the VLANs to which a VSI(s) belongs for egress and ingress checks. This table can point to two lists: one list that is used for ingress VLAN filtering and one used for egress VLAN filtering.

Note: In all previous tables, a VLAN value of zero includes untagged packets.

Note: A port in these tables means either a VSI or the LAN port for transmit packets and a VSI for receive packets.

- Promiscuous modes per VSI:
 - {UPE}— Used to forward to a VSI, all unicast packets.
 - {MPE}— Used to forward to a VSI, all multicast packets.
 - {BAM}— Used to forward to a VSI, broadcast packets.
- {VPE}: Used to forward to a VSI, packets with any VLAN tag.
- Loopback rules:
 - {ALLOWLOOPBACK} Should this port be allowed to send packets to other virtual ports?
 - {PruneEnable} Should this port receive packets it sent? This mode is useful to enable offload of a software switch connected to this virtual port.
- Anti-spoofing parameters. These parameters defines if a packet should be allowed to enter the switch.
 - MAC anti-spoofing enable.
 - VLAN anti-spoofing enable.

The forwarding filters are divided into two groups. Those filters that are needed for tunneled packet formats and those ones that are shared for all packet formats (based on the outer MAC,VLAN). Filters that match tunneled packet formats take precedence on those filters that match all packet formats (as shown in [Figure 7-45](#)).

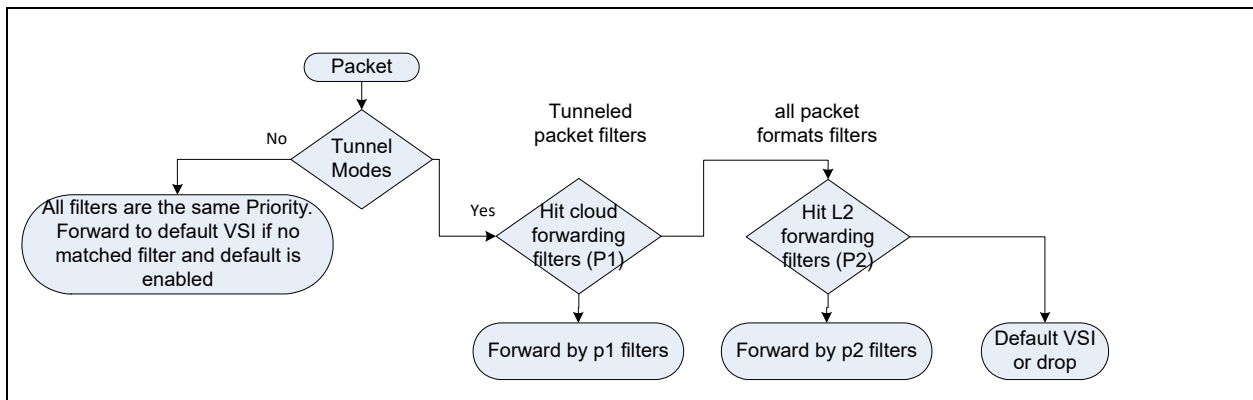


Figure 7-45. Cloud Forwarding Filters Priority

The Add Control Packet Filter ([Section 7.4.9.5.9.9](#)) and Remove Control Packet Filter ([Section 7.4.9.5.9.10](#)) commands are used to add or remove control filters settings. Each control filter can point to a single VSI. There is no support for packet replication based on control filters. An error is returned if an existing control filter is added to point to another VSI.

The Add Cloud filters ([Section 7.4.9.5.9.11](#)) and Delete Cloud filters ([Section 7.4.9.5.9.12](#)) commands are used to control cloud specific filters settings. Each cloud filter can point to a single VSI. There is no support for packet replication based on cloud filters. In addition to those filters, in order for a packet to be received by one of the VSIs, it must also pass the {L2 MAC} filter. In order to enable L2 filtering, the *Flags.Enable L2 filtering* bit in the Add VEB command should be set when creating the cloud VEB.

Note: This L2 filtering is not related to the L2 filtering defined in [Section 7.4.4.3](#).

- The Add MAC, VLAN pair ([Section 7.4.9.5.9.1](#)), Remove MAC, VLAN pair ([Section 7.4.9.5.9.2](#)), Add VLAN ([Section 7.4.9.5.9.3](#)), and Remove VLAN ([Section 7.4.9.5.9.4](#)) commands are used to set regular MAC/VLAN filters.



Note: The Add MAC, VLAN pair command enables filtering based on a MAC address (any VLAN) or on a MAC, VLAN pair. The Add VLAN command is used to define the VLAN membership of ports and is not used to add destination VSIs. Using this method, the VEB can enable MAC-based filtering and promiscuous modes within specific VLANs.

- The Set VSI promiscuous modes (Section 7.4.9.5.9.5) command is used to set promiscuous filters.
- The Add Mirror Rule (Section 7.4.9.5.10.1) and Delete Mirror Rule (Section 7.4.9.5.10.2) commands are used to control mirror rules.

It is assumed that manageability packets and control port packets are not encapsulated.

7.4.4.5.3 Adding L4 Port Filters

The internal switch configuration can be changed to define L4 port filters in place of existing cloud filters by using the Set Switch Configuration command. This change is global for the device and cannot be reversed except by a core reset. An L4 port filter is added using the Big Buffer option of the Add Cloud Filters command, and the L4 port value is placed in bytes 110-111 of the command buffer.

The following modes are supported:

- Mode 0— No L4 port filters; all cloud filters are available. This is the default mode after core reset.
- Mode 1:
 - L4 port filters
 - L4 port
 - Cloud filters removed: None
- Mode 2:
 - L4 port filters
 - {Outer MAC, L4 port}
 - {Outer MAC, Outer VLAN, L4 port}
 - {Destination IP, L4 port}
 - Cloud filters removed
 - Inner MAC
 - {Outer MAC, Tenant ID, Inner MAC}
- Mode 3:
 - L4 port filters
 - {Inner MAC, L4 port}
 - {Inner MAC, Inner VLAN, L4 port}
 - {Destination IP, L4 port}
 - Cloud filters removed
 - Inner MAC
 - {Outer MAC, Tenant ID, Inner MAC}

Note: For Modes 1-3, the L4 port can be specified as the source port or the destination port in the Set Switch Configuration command.

Note: For tunneled packets in Modes 1-3, the destination IP and L4 port values are taken from the inner L3/L4 headers.

7.4.4.5.4 Cloud VEB Flow

Figure 7-46 shows the generic flow of the cloud VEB element. The exact algorithm implementing this flow is described in Section 7.4.8.4.

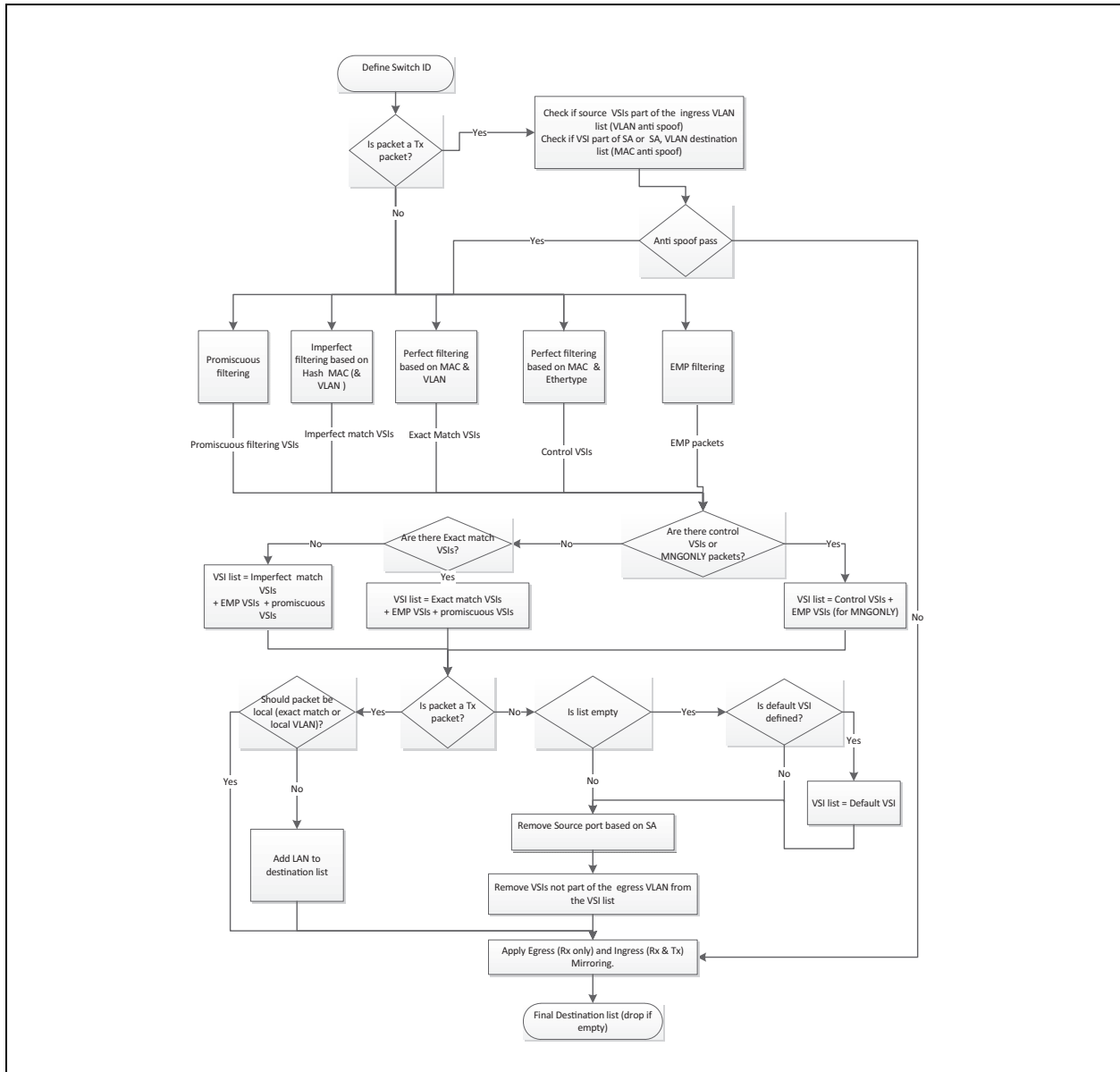


Figure 7-46. Cloud VEB Flow Diagram

7.4.4.6 Port Aggregator

The port aggregators (VEPA) usage model is described in Section 7.4.2.2.1.



7.4.4.6.1 Port Aggregator Switching Rules

The behavior of a port aggregator is the same as the behavior of a VEB. The only difference is that in a port aggregator, loopback should be disabled for all virtual ports (ALLOWLOOPBACK = 0b).

A VEPA can be created using the Add VEB admin command described in Section 7.4.9.5.7.1 and by disabling loopback in the VSIs connected to this VEB.

7.4.4.6.2 Port Aggregator Flow

Figure 7-47 shows the generic flow of the VEPA element. The exact algorithm implementing this flow is described in Section 7.4.8.4.

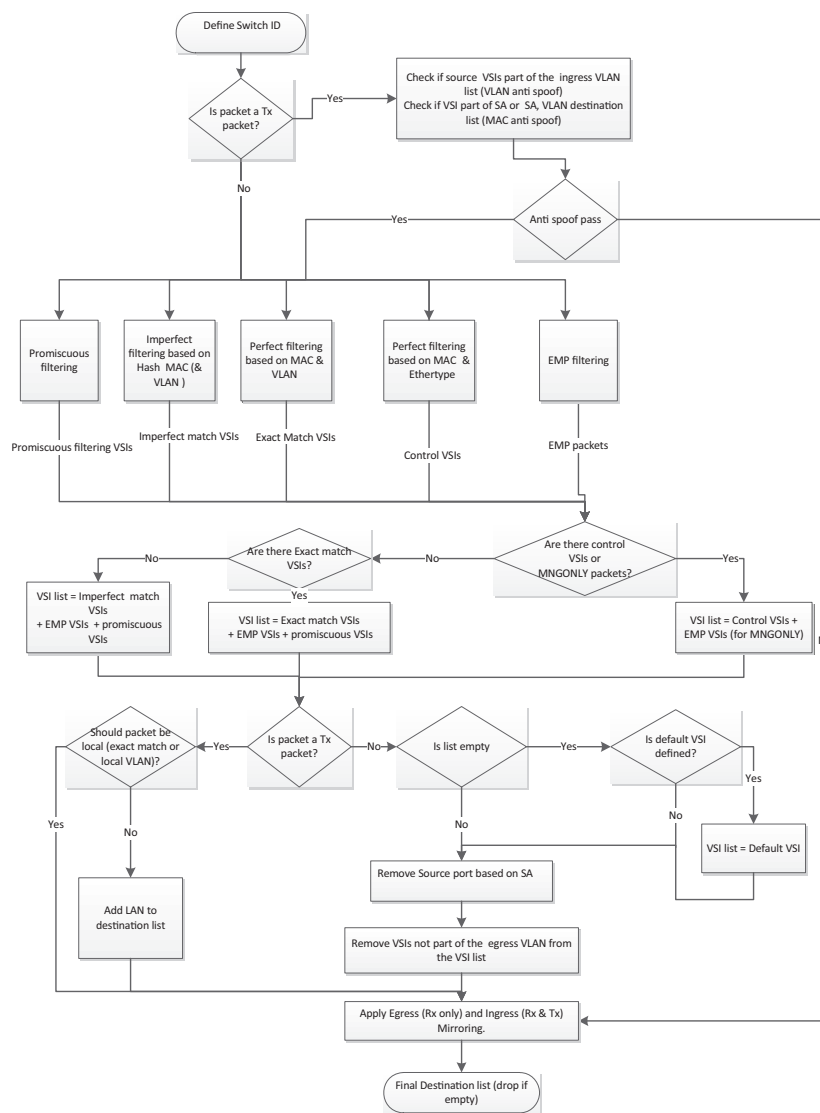


Figure 7-47. VEPA Flow Diagram



7.4.4.7 Floating VEB

A floating VEB is a VEB not connected to the network, enabling only local traffic between the VSI's members of this VEB. This is useful if one of the VSIs acts as a gateway to another network for all the other VSIs within the floating VEB. It can be used to isolate a set of VSIs behind a firewall or to implement NFV functionalities.

A floating VEB is created by setting the *floating VEB* flag and setting a *downlink SEID* of zero in the Add VEB command. The AQ response returns the switch ID used for this floating VEB.

Traffic in a floating VEB is identified using a special S-tag inserted and removed by hardware. Hence, the *Cascaded Port Virtualizer section is valid* bit should be cleared in floating VEB VSIs. Firmware internally sets:

- S-tag = switch ID of the floating VEB
- Switch ID = switch ID of the floating VEB
- S-tag extract mode = 01b
- S-tag insert enable = 1b
- Accept tag from host = 0b

In addition, the *Allow Loopback* flag should be set.

Note: A VSI in a floating VEB might still bypass the switch if enabled via the *SWTCH* flag in the Tx descriptor. A packet sent to the LAN via this mechanism goes out on the VEB with the floating VEB internal S-tag. This is not an expected use case, as only trusted VSIs are allowed to use the *SWTCH* flag.

As VSIs on a floating VEB adds and removes an S-tag, the RXMAX value of the Rx queues in these VSIs should be updated to account for the additional 4 bytes.

Floating a VEB creates an isolated environment for the PF because it cannot completely disconnect the PF from the network. An attempt to call the Delete Element AQ (0x0243) for a VSI that is connected directly to port results in an error.

7.4.4.8 Manageability Sideband Switching

The network interface might be shared between the host and a sideband manageability interface. As described in [Section 9.1.2](#), the sideband interface can be either over a 100 Mb/s Ethernet interface using NC-SI, over SMBus using a legacy pass through mode or NC-SI over MCTP transport or over PCIe using NC-SI over MCTP transport.

The specific filters used to define which packets are forwarded to the sideband interface are described in [Section 9.3](#). This section only describes the relationships between the sideband and other switching elements.

7.4.4.8.1 Network to MC Filtering

The relationships between the sideband filtering and other switching elements follow:

- If no switching elements are defined (such as no VEB or port virtualizer), the sideband decision filters described in [Section 9.3](#) are applied to network traffic. Packets that pass these filters are forwarded to the sideband interface.



- If an S-comp is defined on the port, the sideband might be defined as part of one of the channels in the S-comp. In this case, the sideband decision filters are applied only to the traffic with the matching S-tag. The same S-tag is added to packets sent from the sideband interface.
- The sideband switching is never part of a VEB, port aggregator or port virtualizer and is done in parallel to the forwarding decisions of these elements.

7.4.4.8.2 OS to MC Traffic Filtering

The following rules are used to decide whether operating system traffic should be sent to the sideband interface:

- If no switching elements are defined, for each function associated with a given port, the sideband decision filters associated to this port and applicable to host traffic are applied to the function traffic. Packets that pass these filters are forwarded to the sideband interface. If the sideband interface defines these packets as exclusive (PRT_MNG_MNGONLY is set for this filter), the packets are not sent to the network. Otherwise, they are sent both to the sideband interface and to the network.
- If an S-comp is defined on the port, the sideband might be defined as part of one of the channels in the S-comp. In this case, the sideband decision filters are applied only to host traffic with the matching S-tag. Traffic from the host to the MC is sent to the BMC without the S-tag.

Note: It is assumed that traffic sent from an S-channel different than the S-channel of the MC is forwarded by an external switch.

- If a VEB, port aggregator or port virtualizer is defined, they won't receive packets exclusively sent to the sideband interface.

7.4.4.8.3 MC to OS Traffic Filtering

The following rules are used to decide whether traffic received from the sideband interface should be sent to the host:

- If no switching elements are defined, the port L2 filters are applied to the sideband traffic. Packets that pass these filters are forwarded to the host. Unicast packets that pass exclusive L2 filtering as defined in [Section 7.4.4.3.1](#) are sent only to the host. Multicast or broadcast packets and unicast packets that pass non-exclusive filtering as defined in [Section 7.4.4.3.2](#), are sent both to the host and to the network. Other packets are sent only to the network.
- If a VEB or port aggregator is defined, the switching algorithms of the VEB or port aggregator are applied to the sideband traffic. Packets that pass these filters are forwarded to the host. Unicast packets that pass perfect L2 filtering as defined in [Section 7.4.4.3.2](#) are sent only to the host. Multicast or broadcast packets and unicast packets that pass imperfect filtering as defined in [Section 7.4.4.3.2](#), are sent both to the host and to the network. Other packets are sent only to the network.
- If an S-comp is defined on the port, the sideband might be defined as part of one of the channels in the S-comp. In this case, the switching elements relevant to this channel are applied to the sideband traffic. Traffic from the MC to the host has no S-tag appended.

Note: It is assumed that traffic sent from the S-channel of the MC to a function on a different S-channel is forwarded by an external switch.

- If a port virtualizer is defined on the port or S-channel of the sideband interface, it does not receive traffic from the MC directly. It might still receive such traffic through an external switch.



7.4.4.9 Out of Band Filtering Operation

Part of the type of traffics are also expected to be operational in Sx (equivalent to the D3 or Dr state of the device). The following types of network inbound traffic are included in this category:

- Wake-up per PF.
- Manageability traffic (two channels).
- LLDP traffic (untagged only).

In order to support this traffic, a separate simplified switch mechanism (PF classifier) is provided that is independent of the regular switch and is used to forward packets to these specific locations.

7.4.5 Ports

7.4.5.1 Physical Ports

Physical ports are considered as uplink ports and are not part of the L2 forwarding table. In each switch, there can be one physical port only. All the traffic that is not forwarded to one of the other ports is sent to the uplink egress port. In addition, multicast or broadcast traffic not received from the uplink ingress port is always forwarded to the uplink port, unless it is part of a local VLAN or the switch has no connection to an uplink. Local VLANs are defined in [Section 7.4.6.1.3](#).

7.4.5.1.1 Ethernet Ports

Each of the Ethernet ports (10 Gb/s) can be assigned as an uplink port of a switch if not connected to an S-component. In this case, all the traffic received from this Ethernet port is handed to the switch.

7.4.5.2 VSIs

VSIs are the connections from the switch to entities interfacing with the host. It can be either the entire queue set of a PCIe function or part of the queues of a function.

There can be up to 384 VSIs in the X710-TM4. All the VSIs are equivalent.

At initialization, each PF is assigned a VSI. A default VSI can also be used to provide the VMDq1 or control VSIs functionality described in the sections that follow.

Other VSIs can be assigned to the PF or to VFs and used for the following purposes:

1. VMDq2
2. VMDq1
3. Control ports
4. Mirroring

In addition, EMP VSIs can also be defined. The EMP VSIs are used for:

1. Pass through traffic
2. Control ports

The different types of VSIs and their attributes are described in the following sections.



Mapping of queues to VSIs is described in [Section 7.4.5.2.3](#).

7.4.5.2.1 Host VSIs Types

7.4.5.2.1.1 VMDq2 VSI

The X710-TM4 supports offloading of a VMM software switch. In this mode, the switching between VMs is done using a VEB element or a port virtualizer element. However, all or part of the ports might not be directly connected to the VEB/port virtualizer as SR-IOV functions (VFs) as they might be hidden behind the VMM or a service VM. These VMs can be represented to the VEB as groups of queues in a single PF. There can be up to 256 VSIs for VMDq2.

The only difference between a VMDq2 VSI and a regular VSI associated with the PF is the ability to apply the VM reset flow described in [Section 4.1.2.6](#).

7.4.5.2.1.2 VMDq1-cascaded VEB/S-comp VSI

A VSI might be used for VMDq1 offload. There are two types of VMDq1 offloads:

- A cascaded VEB.
- A cascaded S-comp.

In a cascaded VEB, a software switch is tied to this VSI and uses the MAC, VLAN pairs to queue packets to the right receive queue.

In a cascaded S-comp, the VMDq1 VSI is used to offload a software-based S-comp. A cascaded S-comp VSI is created by setting the *Cascaded Port Virtualizer* flag in the Add VSI command. When an S-channel is used to convey multiple S-tags, it should be connected directly to a VSI; VEB or VEPA elements should not be connected on top of it. Filtering of packets to be sent to a cascaded S-comp VSI should be based only on S-tag and not on MAC, VLAN or other filters.

7.4.5.2.1.3 Control VSI

A single PCI function might be used to control one or multiple switching elements. For example, a single function might be used to control a port virtualizer and a VEB element associated with the function. As a need to separate the control packets of different elements to different queues in receive and identify the switch element that should process the packets in transmit, a separate control VSI should be defined for each element.

A control VSI might receive part or all the link local traffic received by the controlled element. For example, a port virtualizer control VSI should receive LLDP untagged packets. A VEB control VSI behind a port virtualizer gets LLDP packets tagged with the S-tag associated with this VEB. As the control packet's traffic might be handled by a different control port, each control port driver should request forwarding of the relevant packets using the Add Control Packet Filter admin command.

Any VSI can be used as a control VSI as long as the adequate packets are routed to it. A control VSI should have the *Allow Destination Override* flag set to enable it to bypass the switch when sending packets.



At initialization, the control VSI of the MAC is assigned to the EMP. If at a later stage one of the PFs decides to take ownership of this control port, it should assign one of its VSIs as the control port of the MAC. The EMP should be notified of the change using the Stop LLDP Agent command and should disconnect the EMP control port. The PF might elect to add a control port in addition to the EMP control port. If a control port is added to a VEB or a VEPA (either connected to the LAN via a port virtualizer or floating), it does not impact the connectivity MAC control port to the EMP. A control port of a VEB directly connected to the MAC might use the MAC control port or might use a separate VSI. After a port is defined, the owner of the control port (firmware or software device driver) should set the correct filters using the Add Control Packet Filter admin command.

A typical configuration might be:

- Untagged (no S-tag) LLDP packets with a nearest bridge (01-80-C2-00-00-0E) address are forwarded to the control port of the MAC or of a switch element directly connected to the MAC (port virtualizer or VEB) on the EMP. These packets include DCBx or EEE TLVs.
- Untagged (no S-tag) LLDP packets with a nearest non-TPMR (01-80-C2-00-00-03) or nearest customer bridge (01-80-C2-00-00-00) addresses are forwarded to the control port of the switch element directly connected to the MAC (port virtualizer or VEB) on the host. These packets include CDCP TLVs.
- Untagged ECP packets are forwarded to the control port of the switch element directly connected to the MAC (port virtualizer or VEB) on the host. These packets include VDP TLVs.
- S-tagged LLDP packets with a nearest customer bridge (01-80-C2-00-00-00) address are forwarded to the control port of a VEB connected to this S-channel.

Note: A control port can also be used as a default port.

7.4.5.2.1.3.1 Transmission of Packets From a Control VSI

A control VSI might need to send directed multicast packets. For example, sending an LLDP packet with a link local multicast address to the link partner or to one of the VSIs. According to the regular forwarding rules of the switch, such packets are forwarded back to the control port or dropped. To overcome this, the control VSI should set the *SWTCH* field and the optional *DEST_VSI* field in the transmit context descriptor to indicate the desired destination of the packet. See [Section 8.4.2.2.1](#) for details.

7.4.5.2.1.4 Mirroring VSI

A VSI might be used to direct mirror traffic as defined by the mirroring rules in the associated VEB/VEPA. In this case, such a VSI is not expected to receive any traffic apart from the mirrored traffic. Any VSI in the VEB can be set as a mirror port.

A mirror VSI should be set to promiscuous VLAN mode to enable reception packets from any VLAN.

7.4.5.2.1.5 Default VSI

In each port, or each VEB, one VSI can be defined as the default VSI. A port default VSI is available only if a port virtualizer is instantiated on this port, otherwise a VEB default port should be used.

A port default VSI is added by defining the connected VSI port type in the Add Port Virtualizer command to default.

A VEB default VSI is added by defining the downlink VSI port type in the Add VEB command to default.

If a default port is defined, all received traffic within the port/VEB not matching any forwarding rule is sent to the default port; otherwise, it is dropped.



Transmit traffic is not impacted by the default port. Unmatched transmit traffic is sent to the LAN.

A default port is subject to VLAN filtering rules. In order to enable all unmatched traffic to be a default VSI, it should be set to promiscuous VLAN using the Set VSI Promiscuous Modes command.

7.4.5.2.2 EMP VSI Types

7.4.5.2.2.1 Pass Through VSI

A pass through VSI is used to send and receive traffic from an out of band manageability interface. The forwarding rules are described in [Section 9.3](#). If this VSI is located behind a port virtualizer, S-tag insertion and removal for packets sent to or received from the manageability interface is done by the device. VLAN tagging is done by the external MC.

7.4.5.2.2.2 Control VSI

An EMP control VSI behaves as a host control VSI ([Section 7.4.5.2.1.3](#)). It sends and receive packets without S-tag and can send or receive VLAN-tagged or VLAN-untagged packets.

7.4.5.2.3 Mapping of Functions and Queues to VSIs

The function to which each VSI belongs is indicated by the VSI_VSI2F register in the VSI context ([Section 10.2.1.2.13](#)) This register is initiated by the Add VSI admin command ([Section 7.4.9.5.5.1](#)).

Transmit queues are grouped in queue groups as described in [Section 7.8.1.5](#). Each queue group is associated with a VSI, thus creating an association between transmit queues and VSIs. This association is used when applying the VSI policies to transmit traffic.

Receive queues are mapped to VSIs using the VSILAN_QTABLE registers as described in [Section 8.2.1.2](#). These registers are also initiated by the Add VSI admin command.

Mapping can be either contiguous within the function queues or scattered. In the case of scattered mapping, up to 16 queues can be associated with a VSI. Both receive and transmit queues should be associated within the queue set of the VF/PF to which this VSI is associated. So for VSIs associated to a VF, up to 16 queues can be associated with the VSI. For VSIs associated with a PF, all the queues of the PF can be associated with a VSI.

A VSI might choose to queue packets to different receive queues either using RSS and traffic class information or using some other dedicated filters as described in [Section 7.1](#).

7.4.5.2.4 VSI Connections to the Switching Elements

Most VSIs can be connected to the following switching elements as regular ports:

- MAC, VEB, port virtualizer or port aggregator.
- Port virtualizer.

7.4.5.2.5 VSI Context

The context of a VSI contains the following parameters.



Table 7-45. VSI Context

Parameter	Description	Register.Field	Notes
Function Pairing			
Function Type	Defines the owner this VSI can be either a PF, a VF or the EMP.	VSI_VSI2F.FUNCTIONTYPE	00b = VF. 01b = VM. 10b = PF. 11b = EMP.
VF Function Number	The VF number of the function this VSI belongs to.	VSI_VSI2F.VFVMNUMBER	
PF Function Number	The PF number of the function this VSI belongs to.	VSI_VSI2F.PFNUMBER	This field should also be set for VFs.
VSI_ENABLE	Enables transmit and receive from the VSI.	VSI_VSI2F.VS_ENABLE	
Switching Parameters			
SwitchID	Defines the switch to which this VSI belongs. In the case of a VSI connected to a VEB/VEPA/MAC, this should be the same as the uplink switch ID. In the case of a VSI connected to a port virtualizer, this is the S-tag associated with the S-channel of this VSI.	VSI_SRCWCTRL.SWID[11:0], VSI_SRCWCTRL.ISNSTAG, VSI_SRCWCTRL.SWIDVALID	ISNSTAG should be set if the switch ID is not an S-tag.
Enable VLAN Anti Spoof	Check the VLANs used by this VSI to send packets are legitimate.	VSI_SRCWCTRL.VLANAS	
Enable MAC Anti Spoof	Check the SA or SA/VLAN used by this VSI to send packets is legitimate.	VSI_SRCWCTRL.MACAS	
Allow Destination Override	Allows this VSI to set the destination of a packet.	VSI_SRCWCTRL.ALLOWDESTOVERRIDE	
Enable Local Loopback	Allows forwarding to a VSI of packets sent from this VSI.	VSI_RXWCTRL.PRUNEENABLE (inverse logic).	
Enable Loopback	Allows forwarding of packets from this VSI to local VSIs.	VSI_SRCWCTRL.ALLOWLOOPBACK.	
LAN Enable	Defines if the VSI is part of a VEB connected to the network.	VSI_SRCWCTRL.LANENABLE.	
Tag Insertion and Admission Parameters			
Tag Accept Mode	Defines which tags to accept.	VSI_TAR.ACCEPTTAGGED and VSI_TAR.ACCEPT UNTAGGED.	
Port Based Tag	Define the VLAN/S-tag or other tags to insert into a packet.	VSI_TIR.PORT_TAG_ID	
Port Based Tag Insert	Defines whether to use the previous tags.	VSI_L2TAGSTXVALID.PORTBASEDTAGS	
Tag Strip Policy	Defines for each tag, if it should be left in the packet, extracted to the descriptor or removed.	VSI_TSR	
Queue Mapping			
Queue Base	The first queue allocated to this VSI.	VSILAN_QBASE.VSIBASE	
Number of Receive Queues per TC	Defines for each TC how many receive queues are allocated.	VSIQF_TCREGION	
Queue Mapping	Maps 16 virtual queues to physical queues.	VSILAN_QTABLE and VSILAN_QBASE.VSIQTABLE_ENA	

**Table 7-45. VSI Context**

Parameter	Description	Register.Field	Notes
DCB Control			
Enabled UPs	Defines the user priorities used by this VSI.	Implemented in scheduler.	

7.4.5.2.6 VSI Add and Remove Flows

Adding and removing a VSI can only be done by the PF. The following sections describe the flow a PF driver should use to add or remove a VSI.

7.4.5.2.6.1 VSI Initialization Flow

In order to initialize a VSI, software should use the Add VSI command to create the required VSI context. As part of this command, the queues allocated to the VSI should be specified. After this command runs, queues might be initiated as defined in section 8.3.3.1.1.

7.4.5.2.6.2 VSI Disable Flow

VSI's can be disabled either individually or as part of their function. For example, when a VF is disabled, the VF reset stops the VSI's associated with this VF. There are cases where a PF might need to disable a single VSI. The use case for this flow is when a VSI is used as a VMDq interface to a VM that is disabled or moved to another system. In order to enable a quick disable of a VSI, the following flow should be used.

1. Software should stop scheduling packets to the transmit queues assigned to the VSI(s).
2. Stop the VSI traffic:
 - a. For VMDQ VSI's, assert the VM reset using the VSIGEN_RTRIG.VMSWR bit and wait until the VSIGEN_RSTAT.VMRD bit clears. This step stops the traffic in all the queues associated with this VSI.
 - b. For VF's, assert the VF reset using the VPGEN_VFRTRIG.VFSWR bit and wait until the VPGEN_VFRSTAT.VFRD bit clears. This step stops the traffic in all the queues associated with this VF.
3. Stop the relevant queues using the flows described in sections 8.3.3.1.2 and 8.4.3.1.2.
4. The PF polls the *Transactions Pending* flag of the VM, verifying that there are no transaction pending of the VM, as follows:
 - Set the VSI index in the PFPCI_VMINDEX and then poll the PFPCI_VMPEND register.
5. Remove all filters pointing to this VSI/VF.
6. Send a Delete Element admin command with the VSI SEID (for each VSI associated to the VF). This step clears all VSI context values. Remove the VSI from the switch topology and remove it from the scheduler nodes.
7. Wait until the Delete Element command completes.
8. Clear the reset bit (VSIGEN_RTRIG.VMSWR or VPGEN_VFRTRIG.VFSWR).

7.4.6 Advanced Switching Capabilities

7.4.6.1 Security Features

Security features implemented as part of the L2 tag handling such as port-based VLAN and UP handling are described in [Section 7.2.4.4](#) and [Section 7.2.6](#), respectively.

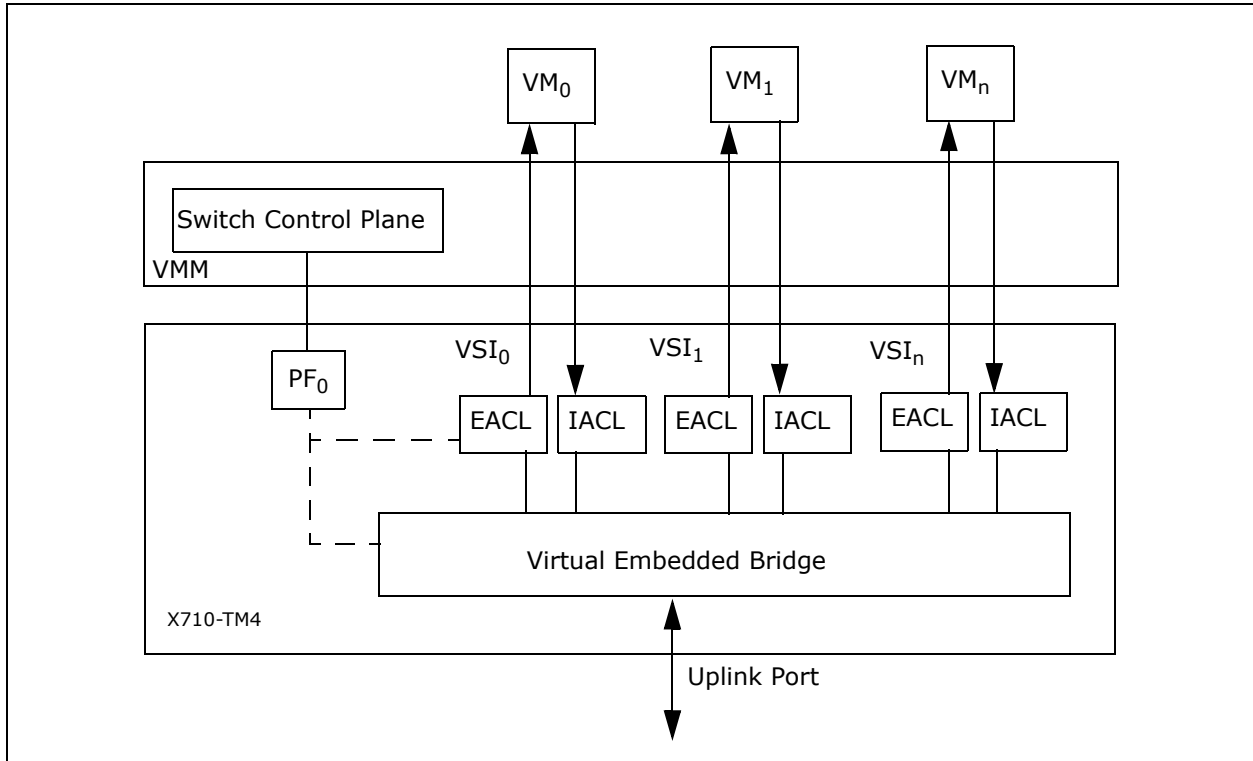


Figure 7-48. Switch Control Plane

7.4.6.1.1 Anti Spoofing

The X710-TM4 supports anti spoofing functionality. This is a security feature that ensures a VM is sending frames with a MAC address and VLAN associated with that VSI. A malicious guest can impersonate as a trusted host by performing MAC address spoofing. If the anti spoofing feature is turned on then only trusted MAC addresses are allowed on the ingress VSI.

7.4.6.1.1.1 MAC Anti Spoofing

A trusted MAC address check is done by doing a MAC address or {VLAN, MAC} lookup on the forwarding database. Since the forwarding database is populated by the VMM or control plane software when the VM/VSI is created, doing a MAC source address lookup on this database ensures that the VSI is transmitting with a source MAC assigned to that VSI.

MAC anti spoofing is enabled by the VSI.Enable MAC anti spoof field in the VSI context.



7.4.6.1.1.2 VLAN Anti Spoofing (Ingress Check)

A VLAN ingress check is done on incoming packets from VMs to ensure if the VSI is a member of the VLAN. This check is done on the ingress VLAN membership table. A VLAN ingress check can be enabled or disabled on each VSI so a VM cannot transmit with a VLAN on which the VSI is not a member. This is applicable for VLAN aware guests. If a port or protocol-based VLAN is configured, a guest is not expected to send packets with 802.1Q tags. As a result, the admit untagged/priority tagged entry has to be set in the port list.

VLAN ingress filtering is enabled by the VSI. Enable VLAN anti spoof field in the VSI context.

7.4.6.1.2 Private VLAN (PVLAN)

The X710-TM4 supports configuring PVLANS in VEB mode. PVLANS are used to provide Layer 2 level security by isolation of VMs within a VLAN (or IP subnet). Since VLAN defines a broadcast domain, all servers connected to the same VLAN can listen to broadcast packets. So a malicious VM can listen to neighboring virtual servers and launch direct attacks bypassing the security policies enforced by enterprise network switches. PVLAN provides Layer 2 security by creating sub-domains within the same primary VLAN without the need for an L3 router. These sub-domains are called secondary VLANs.

A virtual port in a private VLAN can be configured to one of three modes:

- **Isolated Ports** — These cannot talk to any other virtual ports except to ports that are configured as promiscuous ports in the same primary VLAN. So a VM that only needs access to an external network through an uplink port has to be connected to isolated ports.
- **Community Ports** — The ports within a PVLAN community can talk to one another within the same community group (community VLAN) and also to the promiscuous ports in the same primary VLAN. There can be one or more PVLAN communities within the same primary VLAN. So a group of VMs that need to communicate with each other and also need external network access can be configured as members to one of PVLAN communities in the primary VLAN.
- **Promiscuous Ports** — These can talk to all of the ports in the same primary VLAN. Typically uplink ports and virtual ports that are connected to service VMs (for management purposes) can be designated as promiscuous ports.

Note: The meaning of promiscuous ports within the context of PVLAN is different from the generic promiscuous definition.

Figure 7-49 shows a PVLAN configuration.

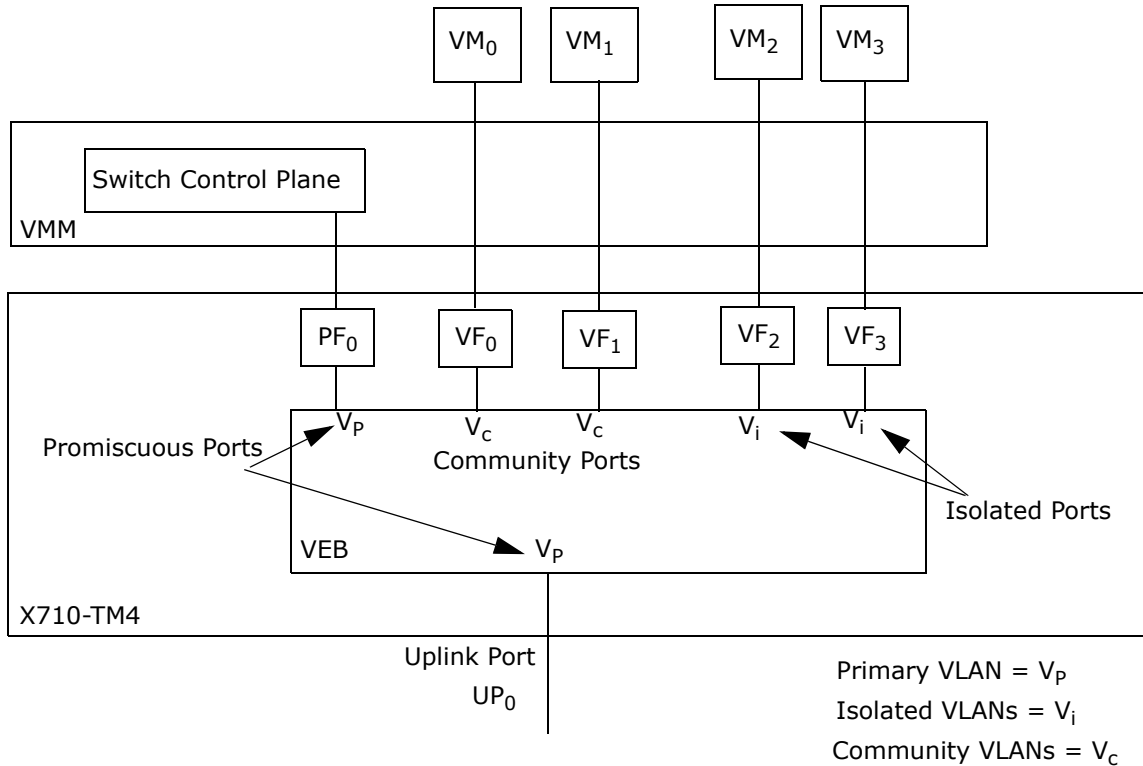


Figure 7-49. Private VLAN Configuration

A primary VLAN V_p is configured on the VEB. For secondary VLANs, V_c is configured as a community VLAN and V_{i1}, V_{i2} are configured as isolated VLANs. The uplink port and the VEB management ports are configured as promiscuous ports. These ports communicate with all the ports in the primary VLAN (V_p) including the community VLAN (V_p, V_c) and isolated VLAN (V_p, V_i).

Community VLAN ports (VF₀ and VF₁) only communicate between themselves and the uplink port UP₀ (and the management port PF₀). Isolated ports VF₂ and VF₃ can only communicate with uplink UP₀ (and the management port PF₀). The isolated ports cannot communicate with other isolated ports (VF₂ cannot communicate with VF₃) or to ports in community VLAN (VF₀ or VF₁).

VLAN pairing is used to represent secondary VLANs. There can be multiple community VLANs configured within a single primary VLAN (for example Vc1, Vc2, etc.); however, only one VLANID Vi is used to represent isolated VLANs.

Virtual port configuration and associated VLAN IDs are created as VLAN pairs as follows:

- Virtual ports (connected to VMs) are configured with VLAN pairs (primary VLAN, secondary VLAN).
- The distinction between the different roles is done by defining the ingress (Tx) and egress (Rx) membership of each VSI for the primary and secondary VLAN as listed in [Table 7-46](#):



Table 7-46. Private VLAN Membership

Port Type	Member of Primary Egress (Rx) List	Member of Primary Ingress (Tx) List	Member of Secondary Egress (Rx) List	Member of Secondary Ingress (Tx) List
Regular VLAN	Yes	Yes	N/A	N/A
Promiscuous	Yes	Yes	Yes	Yes
Community	Yes	No	Yes	Yes
Isolated	Yes	No	No	Yes

7.4.6.1.2.1 Isolated VLAN Configuration and Forwarding

Virtual port VF2, VF3 are configured as isolated ports to be members of both Vp and Vi. Frames from the network to virtual port (ingress) arrives with the primary VLAN ID Vp. However, frames from the virtual port to the network (egress) is always assigned a secondary VLAN ID Vi. Broadcast and multicast frames from the virtual port are not forwarded to other isolated ports in the isolated secondary VLAN Vi. They are only forwarded to uplinks (or ports configured as promiscuous ports in primary VLAN).

The forwarding database has one entry {Vp, MAC2} pairing for virtual port VF2. If a port-based VLAN is assigned to the port, then this is Vi. Otherwise, Vi is the only VLAN ID in the private VLAN to be part of the egress VLAN list for VF2. Packets arriving from the network carry the primary VLAN ID {Vp, MAC2}. Only Vp is part of the ingress VLAN list for this port.

If a unicast packet arrives from the network that is designated as an isolated VLAN it cannot be forwarded to isolated ports and are dropped. If a multicast or broadcast packet arrives from the network with VLAN ID Vi it cannot be forwarded to isolated ports. It can only be forwarded to promiscuous ports and is considered as management port PF0.

7.4.6.1.2.2 Community VLAN Configuration and Forwarding

Virtual port VF0 and VF1 are configured as community ports and are members of both Vp and Vc. Frames from the network to virtual port (ingress) arrive with either primary a VLAN ID Vp or a secondary VLAN ID Vc. However, frames from the virtual port to the network (egress) are always assigned a secondary VLAN ID Vc. Broadcast and multicast frames from the virtual port are only forwarded to other virtual ports that are members of the community VLAN Vc and to uplinks (or ports configured as promiscuous ports in primary VLAN).

The forwarding database has two entries {Vp, MAC1} and {Vc, MAC1} pairing for virtual port VF1. If a port-based VLAN is assigned to this port it is Vc. Otherwise, Vc is the only VLAN ID in the private VLAN to be part of the egress VLAN list for VF2. Ingress packets from the network towards VF2 are used {Vp, MAC1} or {Vc, MAC2} for forwarding. Both Vp and Vc are part of the ingress VLAN list for this port.

If a unicast packet arrives from the network that is designated as community VLAN Vc it can only be forwarded to community ports that are members to the same community VLAN Vc. If a multicast or broadcast packet arrives from the network with a VLAN ID Vc, it can be forwarded to other community ports in the same community VLAN Vc and to promiscuous ports and is considered management port PF0.



7.4.6.1.2.3 Primary VLAN, Promiscuous Port Configuration and Forwarding

A primary VLAN Vp is created with all community ports, isolated ports and promiscuous ports as members. Unicast packets arriving from the network with primary VLAN Vp are forwarded based on {Vp, MACx}. Egress check is done to find out the member ports. Multicast or broadcast packets are forwarded to based on {Vp, Multicast MACx} and forwarded to member ports. Broadcast is forwarded to all members of primary VLAN.

Ports that are configured as promiscuous ports are typically uplink ports (UP0) and switch management ports (PF0). Promiscuous ports are members of primary VLAN Vc and all secondary VLANs, Vc and Viso promiscuous ports can receive broadcast packets from any of the secondary VLANs.

Note: In order to assure proper operation of PVLANS both an internal VEB and external access switch need to be configured with the same PVLAN designations.

7.4.6.1.3 Local VLANs

A VLAN is defined as local. In this case, packets sent with this VLAN ID is forwarded to the network and the packet from the network with this VLAN is dropped.

Locality is defined at the VEB level.

A VLAN can be defined as local using the Add VLAN admin command with the Local VLAN attribute set.

7.4.6.2 Switch Diagnostics

This section describes the mirroring and loopback features. The statistics of the switch are described in [Section 7.11](#).

7.4.6.2.1 Mirroring

The X710-TM4 supports 64 mirroring rules. Each rule can be coupled to any of the VEB or port aggregator in the device. The destination port of each rule (the mirror port) must be connected to the switching element in which the rule is applied. Packets that match any of the mirroring rules assigned to this element is forwarded to the VSI defined in the mirror rule. The setup of a mirror VSI is described in [Section 7.4.6.2.1.3](#).

In order to differentiate packets received according to the different rules, the matched rule is indicated in the Mirror Rule ID (*MIRR*) field in the receive descriptor.

Note: The switching algorithm enables mirroring to mirror ports even if loopback is disabled for this port. The controlling software needs to make sure mirror ports are assigned only to ports that can receive local traffic.

Each rule can is defined by the following parameters:

1. A rule ID — A value in the 0..63 range reflected in the MIRR field in the receive descriptor.
2. A mirror port to which packets matching the mirroring conditions are sent.
3. A set of mirrored ingress VSIs — All packets received by these VSIs are also sent to the mirror port.
4. A set of mirrored egress VSIs — All packets sent by these VSIs are also sent to the mirror port.



Additional rules can be defined to create ingress VLAN mirroring. When such a rule is created, all the traffic received in a set of given VLANs by any of the virtual ports either from the uplink or from local VMs within a given VEB are forwarded to a mirror port.

Note: Packets forwarded by ingress VLAN mirroring or ingress mirroring are not identified by the *MIRR* field and the *UMBCAST* field as a mirror packet. The matched mirror rules can be inferred from the VLAN of the packet. Mirroring according to ingress VLAN mirroring of transmit packets might add an additional copy of the packet to the default VSI of the VEB or the port.

Mirroring rules can be applied using the following admin commands:

- Add Mirror rule (Section 7.4.9.5.10.1).
- Remove Mirror rule (Section 7.4.9.5.10.2).

Note: A mirror port cannot be defined in the ingress mirroring list of another mirroring rule. A single rule can be either ingress VSI, egress VSI or VLAN and cannot include multiple types of rule. Ingress or egress VSI traffic cannot be mirrored to multiple mirrored ports. A VSI cannot be part of more than a single port mirror rule (cannot be mirrored twice). This limitation does not include VLAN mirroring. A control VSI that uses the switch override field (*SWTCH*) in a transmitted packets descriptor cannot be part of an ingress mirror rule.

7.4.6.2.1.1 Ingress Mirroring Flow

Ingress mirroring acts on all the packets that are sent from a given VSI, no matter if the packet is sent to the network or to a local VSI and doesn't depend on the *Allow Loopback* flag in the VSI context.

Packets dropped due to anti spoofing are not mirrored.

Note: If the VSI is connected to a VEPA, ingress mirroring can't be used.

A total of 128 VSI can be mirrored at any given time. If adding any ingress VSI rule causes the total number of mirrored VSI's to exceed 128, an Admin Queue command error code is returned.

7.4.6.2.1.2 Egress Mirroring Flow

Egress mirroring acts on all the packets that are received by a given VSI no matter if the packet was received from the network or from a local VLAN.

7.4.6.2.1.3 Mirror VSI Setup

A VSI used to receive mirror traffic should be dedicated to this type of traffic and should not be used to send or receive other traffic.

7.4.6.2.2 VSI Loopback

Some systems require a local loopback capability at the switch level to enable receiving all the packets sent from a VSI back to the same VSI.



This capability is achieved by using the following steps:

1. When creating the VSI using the Add VSI command, set the *Allow Destination Override* bit.
2. When sending a packet, set in the descriptor the *SWTCH* field to 11b (Target VSI) and set the Segmentation Parameters.VSI field to the value of this VSI.

Packets sent using descriptors with the attributes previously mentioned are sent back to a receive queue of the same VSI.

7.4.6.3 Cascaded Port Virtualizer Offloads

X710-TM4 supports VSIs used as cascaded port virtualizers as described in [Section 7.4.2.4.3](#). Such a VSI provides some special offloads to support this mode.

7.4.6.3.1 Transmit Offloads

S-tag insertion — As this VSI supports multiple S-tags, S-tag cannot be inserted by the X710-TM4 using the port-based tagging mechanism. A software device driver might indicate S-tag to insert in the *L2TAG2* field in the transmit descriptor and set the *IL2TAG2* bit. Hardware then inserts the right tag to the packet.

The ability of a VSI to control S-tag from the descriptor or from the packet is set by the Accept tag from host in the Add VSI command.

Note: If a packet is sent by the software device driver with S-tag as part of the packet, it must also include the VLAN.

If S-tag needs to be inserted, 32-byte transmit descriptors must be used.

7.4.6.3.2 Receive Offloads

- S-tag extraction— As this VSI supports multiple S-tags the software device driver must receive the S-tag in order to process the packet. These tags can be extracted from the packet and stored in the receive descriptor in the *L2TAG2 (1st)* field. The presence of these tags is indicated by the *L2TAG2P* bit. This offload is activated via the *Report S-tag* field in the Add VSI command.

Note: This capability is orthogonal to the VLAN tag extraction and might be requested with or without VLAN extraction.

- Source Pruning — This capability should be disabled in a cascaded S-comp by clearing the *Prune based on ingress E-PID* enable field in the VSI context.
- VMDq1 Queueing — A cascaded port virtualizer might request to queue packets with different tag values to different queues, similar to the VMDq1 mode supported for cascaded VEBs. In this mode, packets with an unrecognized S-tag are forwarded to the default queue of the VSI.



7.4.6.4 DCB and Rate Limit Support

Each VSI can be associated with a scheduling element and with a set of user priorities and traffic classes. Each user priority is represented by a set of queue pairs and a scheduler element. If multiple UPs are associated with the same traffic class, a scheduler element might be added later to group those UPs into a single TC. The VSI scheduling element can be used to define an SLA for this VSI. The UP and TC scheduling elements are used to set an ETS policy for this VSI. Each S-channel created is also associated with a scheduler element enabling a different SLA for each channel.

The assignment of scheduling elements is done internally when an Add VSI or an Add VEB admin command is received. The disassociation is done when a Delete Element admin command is received.

The traffic classes created are set according to the *Enabled UPs* field in the Add VSI command. The handles to the traffic class rate limiters are returned as part of the Add VSI response. The addition of traffic class scheduler elements can be done after the VSI exists using the Configure VSI Bandwidth per Traffic Class admin command ([Section 7.8.4.7](#)).

In order to control rate limiter behavior, the scheduler admin commands ([Section 7.8.4](#)) should be used. When accessing a switching element (VSI or S-channel) scheduler element, the SEID is used as the handle to the node. When accessing a TC the handles returned in the Add VSI response are used.

7.4.7 Flows

This section describes the transmit (Tx) and receive (Rx) processing flows across all switching elements.

7.4.7.1 Tx Flow

The following steps are taken once a packet is accepted for transmission by the switching engine. This flow assumes any stateful offload needed had been applied to the packet:

1. Anti spoofing — If enabled in the Add VSI command, the SA and {SA,VLAN} pair are compared to the forwarding tables of the VEB. If the VSI sending the packet is part of those addresses destination, the packet is allowed to proceed.
2. VLAN insertion — If needed, insert the VLAN tag to the packet, either as a host request or as a port-based VLAN.
3. S-tag insertion — If the VEB or the function is connected to an S-comp, add the S-tag of the channel to the packet.
4. UP replacement — Replace the UP bits as described in [Section 7.2.6.1](#).
5. VEB filtering — If the port/queue that sends the packet is connected to a VEB, the algorithm described in [Section 7.4.8.4](#) is applied to create a list of destination ports. Otherwise, the list of destination ports is simply the network. The VEB filtering for Tx packets includes the following steps:
 - a. Defining the list of VSIs that are a candidate for receiving a packet. These VSIs are defined according to the filters set using the Add MAC, VLAN pair, Add Cloud filters and Add Ethertype filters commands ([Section 7.4.9.5.9](#)).
 - b. Add an EMP owned VSI that should receive the packet ([Section 7.4.4.9](#)). This includes forwarding to the MC channel using decision filters ([Section 9.3](#)) and control packets.
 - c. Potentially removing from this list the VSIs that do not belong to the VLAN group as defined by the Add VLAN command ([Section 7.4.9.5.9.3](#)).



- d. Potentially removing from this list the VSI that sent the packet. This removal can be based on comparison of the SA of the packet to the MAC table.
 - e. Add ingress mirror VSI as defined by the Add Mirror Rules admin command (see [Section 7.4.6.2.1](#)).
6. For a copy of the packet sent to the network follow the instructions in step 7. For a copy of the packet sent to the sideband interface follow the instructions in step 8. For copies of the packet sent to local functions follow the instructions in step 9.
7. Processing packets sent to the network:
 - a. MAC processing — If applicable, insert a CRC in the packet.
 - b. Forward to the network.
8. Processing packets sent to the sideband interface:
 - a. Forward the packet to the sideband interface.
9. Processing packets sent to local functions:
 - a. Forward to function — Forward the packet to the selected function for further processing.

7.4.7.2 Rx Flow

The following steps are taken for each packet received from the network by the switching engine:

1. MAC processing — Check the CRC and other error conditions. If pass bad packets is set (PRT_SBPVSI.SBP), add the bad frames VSI (PRT_SBPVSI.BAD_FRAMES_VSI) to the VSI list. Identify flow control packets and priority flow control packets and react to them (see [Section 3.2.1.5](#)).
2. Untagged packets processing — If the packet is not tagged (no S-tag) and a default switch ID is defined for the port (PRT_SWT_SWITCHID.SWIDVALID = 1b) append a default switch ID to the packet description. The resulting switch ID/S-tag defines the context in which the next step is done.
3. VEB/port aggregator/L2 filter — According to the configured switching element connected to the S-channel/physical port, apply the algorithms defined in [Section 7.4.8.3](#), or [Section 7.4.8.4](#) to define a list of destination ports. This includes the following steps:
 - a. Defining the list of VSIs that are a candidate for receiving a packet. These VSIs are defined according to the filters set using the Add MAC, VLAN pair, Add Cloud filters and Add Ethertype filters commands ([Section 7.4.9.5.9](#)).
 - b. Add an EMP owned VSI that should receive the packet ([Section 7.4.4.9](#)). This includes forwarding to the MC channel using decision filters ([Section 9.3](#)) and control packets.
 - c. Potentially removing from this list the VSIs that do not belong to the VLAN group as defined by the Add VLAN command ([Section 7.4.9.5.9.3](#)).
 - d. Potentially removing from this list the VSI that sent the packet. This removal can be based on a comparison of the SA of the packet to the MAC table.
 - e. Add egress mirror VSIs as defined by the Add Mirror Rules admin command (see [Section 7.4.6.2.1](#)).
4. The resulting list of destination ports can include local ports or sideband interface.
5. For a copy of the packet sent to the sideband interface follow the instructions in step 6. For copies of the packet sent to local functions follow the instructions in step 7.
6. Processing packets sent to local functions:
 - a. S-tag and VLAN removal — If applicable, remove the S-tag and VLAN tag from the packet and optionally report them in the Rx descriptor.
 - b. UP replacement — Replace the UP bits as described in [Section 7.2.6.2.1](#).



- c. Forward to function — Forward the packet to the selected function for further processing including queuing.
7. Processing packets sent to the sideband interface or to the EMP:
- a. Forward the packet to the sideband interface or to the EMP. The VSI context defines the destination within the EMP VSI (one of the sideband interfaces or the EMP) either using the regular switch or the packet classifier.

7.4.8 Switching Algorithms

7.4.8.1 LAN/SAN Port Extender Algorithm

This algorithm translates a destination MAC address to a switch ID. After this algorithm is applied, the regular per switch forwarding algorithm should be applied to each of the switch IDs. The algorithm is relevant only to receive traffic.

```
// Global parameters
Port.DefaultStag; Can be Null or an S-tag.

// Define Lookup tables
DA Table: Array of {DA ,LAN , Switch ID}

EComp_function(Packet) {
// Variables
    STag = Port.DefaultStag
    SourceStagFlag = FALSE // SourceStagFlag is not used by this algorithm.

//Define packet parameters
    DA = Packet.DA: the DA of the packet.
    Source Port // The port from where the packet was received.

// Switching algorithm
// Selection of S-tag using DA
    If ({DA, source port} match entry e in DA table): DStag = e.S-tag;
    Return DStag, SourceStagFlag;
}
```

7.4.8.2 S-comp Forwarding Algorithm

The following algorithm refers only to S-comp receive traffic. As previously described, transmit traffic is always sent to the LAN.

Packets with unmatched S-tag is forwarded to a default VSI of the port, if enabled.

Packets forwarded to the default VSI cannot be subject to L2 filtering.



This algorithm defines a switch ID that identifies the S-channel to which the packet should be forwarded. The switch ID is used to identify a potential VEB connected to this S-channel. The forwarding inside this channel is defined according to the L2 forwarding algorithm ([Section 7.4.8.3](#)) or VEB/VEPA algorithm ([Section 7.4.8.4](#)).

If no VEB is defined, the VSI connected to the S-channel should be put in promiscuous mode using the Set VSI Promiscuous Modes command in order to receive all the traffic with the given S-tag.

If an S-comp is not defined on the port, all packets with an S-tag are dropped.

```
// Global parameters

Port.DefaultVSI; Can be Null or a VSI

Port.Default_Switch_ID; // The switch ID to use for untagged.

DefaultVSIVValid - per port// If set, packets with an S-tag that do not match any of these
programmed S-tags or an untagged packet that do not match the expected ethertypes will be sent to
the default VSI, otherwise the packet is dropped.

// Define Lookup tables

StagTable: Array of {STAG}

// Control Port forwarding rules

SComp_function(Packet) {

// Variables

Dest_VSI - Null; // Destination VSI in case of default VSI;

Switch_ID = Null;

//Define packet parameters

STag = Packet.Stag: the S-Tag of the packet.

Untagged // True if packet has no S-tag.

// Switching algortihm

// control port forwarding

// Note - the usual case of a control VSI is to receive LLDP packets or other packets based on
special Ethertypes. However, the device allows any L2 forwarding to the control VSI assuming the
right forwarding rule is added. See Section 7.4.8.4 for details of the available rules.

If (Untagged) { Switch_ID = Port.Default_Switch_ID;}

// S-tag forwarding

If(Stag match entry e in StagTable) {

    Switch_ID = S-tag

} else if (DefaultVSIVValid)

    Dest_VSI = Port.DefaultVSI; Switch_ID = Port.Default_Switch_ID; // The regular VEB algorithm
of the default switch ID VEB will apply to this packet.

else drop packet.

Return Switch_ID, Dest_VSI;

}
```




7.4.8.3 L2 Filtering Algorithm

The following pseudo code describes the algorithm used to determine if a packet passes the L2 filtering element.

```

// Global parameters

// Define Lookup tables
MacVlan Table: Array of {MAC (MAC Address), VLAN (VLAN tag)}
Mac Table: Array of {MAC (MAC Address)}
VLAN table: Array of {VLAN (VLAN tag)}
EtherType Table : Array of {Etype (Ethertypes Values)}
MacEtherType Table: Array of {MAC (MAC Address), Etype (Ethertype value)}

// Define Virtual ports modes
Port.PUE // Promiscuous Unicast Enable
Port.PME // Promiscuous Multicast Enable
Port.BAM // Broadcast Enable
Port.VPE // Promiscuous VLAN enable
Port.MaxSize: Max Packet size

L2_function(Packet)

// Variables
MFilter: = False // MAC Filtering
VFilter: = False // VLAN Filtering
EFilter: = False // Exclusive Filtering
NEPMFilter = False // Non Exclusive Perfect MAC Filtering
NEPFilter = False // Non Exclusive Perfect Filtering
NEIPMFilter = False // Non Exclusive Imperfect MAC Filtering
NEIPFilter = False // Non Exclusive Imperfect Filtering
Pass = False // Final decision.

//Define packet parameters
DA = Packet.DA //Destination Address of the packet
VID = Packet.VLAN ID // Vlan tag of the packet
Etype = Packet.Ethertype // Ethertype of the packet
AddressType //Type of address of the DA. Can be Unicast, Multicast or Broadcast.
HDA = HashFunction(DA) .

PSize: Packet size. This do not include any tag or other header removed by previous stages or to be
added by following stages.

```



```
// Exclusive Filters
For Each entry e in MacVlan Table
    If (DA == e.MAC and (VID == e.VLAN or (VID == NULL and e.VLAN == 0)) MFilter = True;
For Each entry e in Mac Table
    If (DA == e.MAC) MFilter = True;
For Each entry e in Ethertype Table
    If (Etype == e.Etype) MFilter = True;
For Each entry e in MacEthertype Table
    If (DA == e.MAC and Etype == e.Etype) MFilter = True;
For Each entry e in Vlan Table
    If (VID == e.VLAN or (VID == NULL and e.VLAN == 0)) VFilter = True;
// VLAN filters are ANDed with the previous filters unless promiscuous VLAN is enabled
EFilter = MFilter and (VFilter or p.VPE) and AddressType == Unicast;

// Non Exclusive Perfect Filters
If (AddressType == Unicast and p.PUE == 1) NEPMFilter = True;
If (AddressType == Multicast and (p.PME == 1 or MFilter)) NEPMFilter = True;
If (AddressType == Broadcast and (p.BAM == 1 or MFilter)) NEPMFilter = True;
// VLAN filters are ANDed with the previous filters unless promiscuous VLAN is enabled
NEPFilter = NEPMFilter and (VFilter or p.VPE);

// Non Exclusive Imperfect Filters
For Each entry e in HashMACVlan Table
    If (HDA == e.HashMAC and (VID == e.VLAN or (VID == NULL and e.VLAN == 0)) and AddressType ==
e.AddressType) NEIPMFilter = True;
For Each entry e in HashMAC Table
    If (HDA == e.HashMAC and AddressType == e.AddressType) NEIPMFilter = True;
// VLAN filters are ANDed with the previous filters unless promiscuous VLAN is enabled
NIEPFilter = NIEPMFilter and (VFilter or p.VPE);
// Packet size filtering is done at the queue level, so it is not part of the switch algorithm
Pass = (EFilter or NEPFilter or NEIPFilter)
Return Pass;
}

HashFunction(MAC) {
Hash = MAC[6:0] // Use the 7 least significant bits of the MAC address (last bits on the wire).
Return Hash
```



```
}

```

7.4.8.4 Cloud VEB Algorithm

```
// Global parameters

// Define Lookup tables

// L2 forwarding tables

MacVlan Table: Array of {MAC , VLAN , DestinationVSIList };

MAC Table: Array of {MAC , DestinationVSIList }

VLAN table: Array of {VLAN, IngressVSIList, EgressVSIList, Local}

HashMacVlan Table: Array of {HashMAC (Hash Values), VLAN, AddressType (unicast/Multicast),
DestinationVSIList }

HashMac Table: Array of {HashMAC (Hash Values), AddressType (unicast/multicast),
DestinationVSIList }

// Control packets forwarding tables

MacEthertype Table : Array of {MAC , Etype , Source, DestinationVSIList, Drop}

Ethertype Table : Array of {Etype , Source, DestinationVSIList, Drop}

// Cloud Forwarding tables

OuterIP: Array of {Outer IP, DestinationVSIList}; // Outer IP is valid only for encapsulated
packets (Filter type = 0x1).

InnerMAC_InnerVLAN: Array of {Inner MAC, InnerVLAN, DestinationVSIList}; // This table can be
matched by packets with NVGRE, VXLAN or Geneve tunneling (Filter type = 0x3).

InnerMAC_InnerVLAN_Key: Array of {Inner MAC, InnerVLAN, Key, EncapsulationType,
DestinationVSIList}; // This table can be matched by packets with NVGRE, VXLAN or Geneve tunneling
(Filter type = 0x4).

InnerMAC_Key: Array of {Inner MAC, Key, EncapsulationType, DestinationVSIList}; // This table can
be matched by packets with NVGRE, VXLAN or Geneve tunneling (Filter type = 0x6).

InnerMAC: Array of {Inner MAC, DestinationVSIList}; // This table can be matched by packets with
NVGRE, VXLAN or Geneve tunneling (Filter type = 0xA).

InnerMAC_OuterMAC_Key: Array of {Inner MAC, OuterMAC, Key, EncapsulationType, DestinationVSIList};
// This table can be matched by packets with NVGRE, VXLAN or Geneve tunneling (Filter type = 0xB).

InnerIP: Array of {Inner IP, DestinationVSIList}; // Inner IP is valid for all packets. In non
encapsulated packets, it is the sole IP (Filter type = 0xC).

Promiscuous_List: Array of {VLAN, Address_type DestinationVSIList}; // Promiscuous modes per VLAN

// Define VSI modes

VSI[i].PUE: Promiscuous Unicast Enable per VSI (promiscuous VLAN only)// Target VSI

VSI[i].PME: Promiscuous Multicast Enable per VSI (promiscuous VLAN only)// Target VSI

VSI[i].BAM: Broadcast Enable per VSI (promiscuous VLAN only)// Target VSI

VSI[i].VLANPruneEnable // Target VSI - Should be cleared in case of promiscuous enable

```



```
VSI[i].AllowLoopback: Loopback Enable per VSI // Source VSI

VSI[i].MACPruneEnable: // Defines if the source VSI (identified either by SA,VLAN or by SA) should
be removed from the list of destination VSI.

VSI[i].EnableMACAntiSpoof: // Source VSI

VSI[i].EnableVLANAntiSpoof: // Source VSI

VSI[i].LANEnable // Source VSI - should be set in each VSI tied to a switch that is connected to
the LAN. (VSI_SRCCTRL.LANENABLE). This bit is cleared if the VSI is added to a floating VEB.

VSI[i].AllowOverride // Allows override of the destination - usually set for control VSIs.

// Switch generic parameters

DefaultVSI; Can be Null or one of the VSIs. // Defined as part of the S-tag table.

ControlVSI; Can be one of the VSIs or the embedded controller.

// Mirroring rules

Int n_mirror: Number of mirror rules

enum {ALL_EGRESS, ALL_INGRESS, VSI_EGRESS, VSI_INGRESS, VLAN_MIRROR} RULE_TYPE;

Mirror rule[1 .. n_mirror] = {RULE_TYPE type, IngressVSIList (List of VSI), EgressVSIList (List of
VSI), VLANList ( List of VID), MirrorPort};

Dport Switch_function(SVSI : Source VSI, Packet, FromLAN, FromHost)

// Variables

CDport: List of VSI = {} // Cloud Filtering - empty list at startup.

PFDPort: List of VSI = {} // Perfect Filtering - empty list at startup.

IFDPort: List of VSI = {} // Imperfect Filtering - empty list at startup.

PMDport: List of VSI = {} // Promiscuous Filtering - empty list at startup.

OFDPort: List of VSI = {} // Override VSI Filtering - empty list at startup.

VFDPort: List of VSI = {} // VLAN membership VSI Filtering - empty list at startup.

MNGPort: List of VSI = {MDEF filtering result as defined in Section 9.3}

MNGOnly: Bool = {MNGonly result as defined in Section 9.3}

DPort: List of VSI = {} // Final List - empty list at startup. Can include also the LAN port for Tx
packets.

PassAntiSpoof = False;

PassMACAntiSpoof = False;

PassVLANAntiSpoof = False;

//Define packet parameters

DA = Packet.DA: Destination Address of the packet

SA = Packet.SA: Source Address of the packet

VID = Packet.VLAN ID: Vlan tag of the packet - equal to zero if untagged packet.

Etype = Packet.Ethertype: Ethertype of the packet
```



AddressType: Type of address of the DA. Can be Unicast, Multicast or Broadcast.

HDA = HashFunction(DA).

OuterIP = Packet.OuterIP // Relevant only for encapsulated packets.

InnerIP = Packet.InnerIP // Inner IP is valid for all packets. In non encapsulated packets, it is the sole IP.

InnerMAC = Packet.InnerMAC;

InnerVLAN = Packet.InnerVLAN;

Key = Packet.NVGRE TNI key or Packet.VXLAN VNI key or Packet.Geneve VNI Key;

EncapsulationType = NVGRE, VXLAN Geneve;

PSize: Packet size. This do not include any tag or other header removed by previous stages or to be added by following stages.

LocalVLAN = False: Define if the packet is part of a local VLAN. See below for calculation.

Destination: // Can be either uplink (send only to LAN), local (switch decides on destination, but do not send to LAN), switchBased (regular switching algorithm) or a TargetVSI. This is a descriptor bit for packets received from a control port allowing override of the switching decision. This may be useful when sending switch generated packets like DCBX packets. Relevant only for transmit packets

Source = Tx or Rx // defines if this is a packet sent by the X710-TM4 or received by the X710-TM4.

Bool Promiscuous_Joins_Exact;//Set by the Set Switch Configuration AQC

// Combine override destination option.

// This logic creates the following indications to be used later

Specific_Port = NULL; // A specific port defined as the sole destination of the packet.

SwitchToLAN = TRUE; // Allow forwarding to LAN

SwitchToLocal = TRUE; // Allow forwarding to local VSI.

if (Destination == TargetVSI and SVSI.AllowOverride) {Specific_Port = Target VSI from Descriptor};

if ((Destination == Uplink and SVSI.AllowOverride) {Specific_Port = LAN};

if ((Destination == Local and SVSI.AllowOverride) or SVSI.LANDisable == TRUE) SwitchToLAN = FALSE;

If (FromLan) SPort = LAN else SPort = SVSI.

// Anti Spoofing

// Local VLAN calculation

For each v in VLAN Table {if (v.VLAN == VID and v.LocalVLAN == True) LocalVLAN = TRUE}

// Need to check if the packet can enter the switch before applying all the rules

If (SVSI.EnableMACAntiSpoof == True and FromHost) {

// Only perfect match filters are used for anti spoofing checks

For Each entry e in MacVlan Table

if (SA == e.MAC and (VID == e.VLAN or (VID == NULL and e.VLAN == 0)) and SVSI in e.DPortList) PassMACAntiSpoof = True



```
For Each entry e in Mac Table
    if (SA == e.MAC and SPort in e.DestinationVSIList ) PassMACAntiSpoof = True
    } else PassMACAntiSpoof = True;
// The step below (VLAN anti spoof) is really ingress VLAN filtering
If (SPort.EnableVlanAntiSpoof == True and FromHost) {
    For Each entry e in VLAN Table
        // Note - an untagged packet will be compared with an entry of the table with VLAN = 0.
        if (VID == e.VLAN and SVSI in e.IngressVSIList) PassVLANAntiSpoof = True;
    } else if (FromLAN and LocalVLAN == True) {
        PassVLANAntiSpoof = FALSE;
    } else PassVLANAntiSpoof = True;
    PassAntiSpoof = PassVLANAntiSpoof and PassMACAntiSpoof;
If (PassAntiSpoof == False) {DPort = null; Goto Mirroring Rules }// Drop Packet as DPort is empty;
// Switching algorithm
// Perfect Filters
For Each entry e in MacVlan Table {
    If (DA == e.MAC and (VID == e.VLAN or (VID == NULL and e.VLAN == 0)))
        PFDPort = PFDPort + e.DestinationVSIList; // Add VSI matching the MAC, VLAN pair.
    }
For Each entry e in Mac Table { :
    If ((DA == e.MAC )
        PFDPort = PFDPort + e.DestinationVSIList; // Add ports matching the MAC only entry.
    }
// Imperfect Filters
    For Each entry e in HashMACVlan Table where (HDA == e.HashMAC and (VID == e.VLAN or (VID ==
NULL and e.VLAN == 0 and AddressType == e.AddressType))):
        IFDPort = IFDPort + e.DestinationVSIList; // Add VSIs matching the HashMAC, VLAN pair.
    For Each entry e in HashMAC Table where (HDA == e.HashMAC and AddressType == e.AddressType):
        IFDPort = IFDPort + e.DestinationVSIList; // Add VSIs matching the HashMAC.
    }
// Apply promiscuous modes
For each port p in VEB {
    If (AddressType == Unicast and p.PUE == 1) PMDPort = PMDPort + p;
    If (AddressType == Multicast and p.PME == 1) PMDPort = PMDPort + p;
```



```

    If (AddressType == Broadcast and p.BAM == 1) PMDPort = PMDPort + p;
}

// Apply per VLAN promiscuous modes
For Each entry e in Promiscuous_List Table where (VID == e.VLAN and AddressType == e.AddressType):
    PMDPort = PMDPort + e.DestinationVSIList; // Add VSIs matching the promiscuous list.

// Cloud filters
// Application Source-IP, Inner MAC filtering. Filter 0xD in Add Cloud Filters command
For Each entry e in SourceIpInnerMAC Table where (InnerMAC == e.InnerMAC and SourceIP ==
e.SourceIP):
    CDPort = CDPort + e.DestinationVSI;

// Inner MAC/VLAN/Key filtering. Filter 0x4 in Add Cloud Filters command
For Each entry e in InnerMAC_InnerVLAN_Key Table where (InnerMAC == e.InnerMAC and InnerVLAN ==
e.InnerVLAN and Key == e.Key_Key Low and EncapsulationType = e.EncapsulationType):
    CDPort = CDPort + e.DestinationVSI;

// Inner MAC filtering. Filter 0xA in Add Cloud Filters command
For Each entry e in InnerMAC Table where (InnerMAC == e.InnerMAC):
    CPort = CPort + e.DestinationVSI;

// Inner MAC/Key filtering. Filter 0x6 in Add Cloud Filters command
For Each entry e in InnerMAC_VLAN_Key Table where (InnerMAC == e.InnerMAC and Key == e.Key and
EncapsulationType = e.EncapsulationType):
    CDPort = CDPort + e.DestinationVSI;

// Inner MAC/VLAN filtering. Filter 0x3 in Add Cloud Filters command
For Each entry e in InnerMAC_VLAN Table where (InnerMAC == e.InnerMAC and InnerVLAN ==
e.InnerVLAN):
    CPort = CPort + e.DestinationVSI;

// Outer MAC/Key/Inner MAC filtering. Filter 0xB in Add Cloud Filters command
For Each entry e in InnerMAC_OuterMAC_Key Table where (InnerMAC == e.InnerMAC and OuterMAC ==
e.MAC and Key == e.Key_Key Low and EncapsulationType = e.EncapsulationType):
    CPort = CPort + e.DestinationVSI;

// Inner IP filtering. Filter 0xC in Add Cloud Filters command
For Each entry e in IP Table where (InnerIP == e.InnerIP):
    CDPort = CDPort + e.DestinationVSI;

// L2 filtering rule
// Checks that the packet source MAC address was added by one of the MAC address based filters
(Either initial PF MAC address, Add MAC VLAN AQ command, Add Ethertype AQ command or filer 9 in Add
Cloud Filters).

```



```
if (DA does not match any entry in MAC table and L2 filtering is enabled) CDPort = NULL;

// Override filters
For Each entry e in Ethertype Table where (Etype == e.Etype and Source == e.Source):
    OFDPort = OFDPort + e.DPortList; // Control port filtering.
For Each entry e in MacEthertype Table where (DA == e.MAC and Etype == e.Etype and Source ==
e.Source):
    OFDPort = OFDPort + e.DPortList; // Control port filtering.

// Create unified list
If (MngOnly) DPort = MNGPort;
Else If (OFDPort not empty) Dport = OFDPort + MNGPort; // Override
Else if (CPort not empty and Promiscuous_Joins_Exact) DPort = CPort + PMDPort; // cloud &
promiscuous
Else if (CPort not empty and !Promiscuous_Joins_Exact) DPort = CPort; // cloud only
Else if (PFDPort not empty and Promiscuous_Joins_Exact) DPort = PFDPort + PMDPort + MNGPort; //
exact & promiscuous
Else if (PFDPort not empty and !Promiscuous_Joins_Exact) DPort = PFDPort + MNGPort; // exact only
else DPort = IFDPort + PMDPort + MNGPort; //imperfect & promiscuous
Dport_for_mirroring = Dport;
if (DPort is empty and FromLAN and DefaultVSI != Null)
    DPort = DefaultVSI;
// Fall back to PV default port if no match to any filter.
if (Dport is empty and PortDefaultVSI != Null and FromLAN) DPort = PortDefaultVSI;
DPort_for_mirroring = Dport; // mirroring ignores VLAN pruning

// VLAN egress Filtering
// Note - to use VLAN only filtering (ignore all MAC addresses), the UPE, MPE and BAM bits should be
set in all ports. See Section 7.4.6.1.2 for rules to set the egress and ingress VLAN lists
For each entry p in DPort {
    if (p.VLANPruneEnable == True) {
        If (VID match VLAN in VLAN table) {
            for each e in Egress Vlan Table where (VID == e.VLAN or (VID == NULL and e.VLAN == 0))
                if (p NOT in e.EgressVSIList) DPort = DPort - p; // prune ports which are not member
of the VLAN;
        }
        Else Dport = Dport - p; // Remove all ports which are not in Promiscuous VLAN if VLAN not
found in VLAN table.
    }
}
```




```

}

// Add external ports if needed
Bool May_loopback_for_mirror = FALSE;

// Note: In VEPA mode, all Tx packets should go to LAN only (VSI[i].AllowLoopback == 0)
if (
    (PFDport is empty or AddressType == Multicast or AddressType == Broadcast or
    VSI[i].AllowLoopback == 0) and
    (CPort is empty or VSI[i].AllowLoopback == 0) and
    OFDPort is empty and FromHost and LocalVLAN == FALSE and SVSI.EnableLAN) {

    DPort = Dport + LAN port;

    If (DPort = LAN Port) May_loopback_for_mirror = TRUE;

    DPort_for_mirroring = Dport_for_mirroring + LAN port;
}

// Pruning Rules
// Control Packets drop rules.
PacketDropped = FALSE

For Each entry e in Ethertype Table where (Etype == e.Etype and Source == e.Source):
    if (e.Drop == True) { DPort = Null;DPort_for_mirroring = NULL; PacketDropped = TRUE}

    For Each entry e in MacEthertype Table where (DA == e.MAC and Etype == e.Etype and Source ==
    e.Source):

        if (e.Drop == True) { DPort = Null;DPort_for_mirroring = NULL; PacketDropped = TRUE}

// Find the source port based on the source MAC address.

// For Each entry e in Mac Table not pointing to a list // unicast addresses are pointing to a
single VSI

    if (SA == e.MAC ) SourcePort = e.MAC;

For Each entry e in MacVLAN Table not pointing to a list // unicast addresses are pointing to a
single VSI

    if (SA == e.MAC and VLAN = e.VLAN ) SourcePort = e.MAC;

if (SVSI.ALLOWLOOPBACK == FALSE) SwitchToLocal = FALSE;

// Note - switch to local may be also disabled via a specific port setting to LAN, but this is
handled below in the Specific_port handling.

For Each port p in Dport {

    // Remove ports from list if loopback disabled for this port

    if (From Host and SwitchToLocal == FALSE and p != LAN Port) {

        DPort = Dport - p;

        DPort_for_mirroring = DPort_for_mirroring - p;

    }
}

```



```
// Remove uplinks from list if requested by descriptor
if (p == LAN Port and SwitchToLAN == FALSE) {
    Dport = Dport - p ;
    DPort_for_mirroring = DPort_for_mirroring - p;
}
// Prune Source port
if (p.MACPruneEnable== True and ( p = SourcePort or p == Source VSI)
    {
        DPort = Dport - p;
        DPort_for_mirroring = DPort_for_mirroring - p;
    }
// Prune by size is done at the queue level, so this is not handled in the switch.
}
}
}
// Override switch decision according to descriptor overrides.
If (Specific_Port != NULL) {
    DPort = Specific_Port;
    DPort_for_mirroring = NULL;
}
// Mirroring rules
MatchedRuleID = Null;
for ( i = 1 to n_Mirror) { // Apply all mirroring rules
// Note - a port can be member only in a single VSI based mirror rule.
    Bool Mirror = FALSE;
    Bool VLAN_Mirror = FALSE;
If(FromHost) {
    Case Mirror rule[i].type {
        ALL_EGRESS {
            If (DPort_for_mirroring not empty) Mirror = True;
        };
        ALL_INGRESS {
            If (Specific_port == NULL or Specific_port == LAN) Mirror = True;
        };
    };
};
```



```

VSI_INGRESS {
    // This rule is applied even if packet is dropped.
    if (SPort in Mirror rule[i].IngressPortList and (Specific_port == NULL or Specific_port ==
LAN)) Mirror = True;};

VSI_EGRESS {
    For each p in Dport_for_mirroring {
        if (p in Mirror rule[i].EgressPortList) Mirror = True;
    }
};

};

If (FromLan) {
    Case Mirror rule[i].type{
        ALL_EGRESS {
            Mirror = True;
        };
        VSI_EGRESS {
            For each p in Dport_for_mirroring {
                if (p in Mirror rule[i].EgressPortList) Mirror = True;
            }
        };
    };
};

if (Mirror rule.type = VLAN_MIRROR and VID in Mirror rule.VLANList and Specific_port == NULL
and PacketDropped = FALSE and (SVSI.ALLOWLOOPBACK or FromLAN)) VLAN_Mirror = True;

if (Mirror) {
    DPort = Dport + Mirror rule[i].MirrorPort;
    MatchedRuleID = i; // Only a single non VLAN rule should match
}

if (VLAN_Mirror) {
    DPort = Dport + Mirror rule[i].MirrorPort;
    If (May_loopback_for_mirror) DPort = Dport + DefaultVSI/PortDefaultVSI;
}

}

Return Dport // Note - the Dport list, should include a single copy of each VSI.
}

```



7.4.9 Switch Programming Model

7.4.9.1 Switching Structure Representation

This section describes the model used to represent the switching elements.

The objects represented in the models are identified by a 10-bit, device wide switch element identifier (SEID). The SEID of an element is returned after creating the element and should be referenced when creating ties between elements. The SEID of elements created automatically can be retrieved using the Describe Structure command.

The objects represented can be either external or internal to the switch. Possible external elements are:

Table 7-47. External Elements

External Element	Abbreviation	Element Type	Description
Physical Port MAC	MAC	1	This element is always present. An SEID is allocated at power on for each enabled port.
Physical Functions	PF	2	PF elements is created at PCIe reset for each enabled PF.
Virtual Functions	VF	3	VF elements are created when SR-IOV is enabled on a physical function.
Embedded Micro Processor	EMP	4	The embedded micro processor can be used as a control port for the different switching elements. All the ties to the EMP should be explicitly created when creating an element. An EMP can contain multiple EMP queues that are only used to represent logical connections to the switch. This is used to enable multiple control ports in the EMP. This element includes the MC interface used to enable pass through traffic to an external MC via an out of band connection.

The internal elements are elements that are used as part of the switching decision. The following elements are available:



Table 7-48. Internal Elements

Internal Element	Abbreviation	Element Type	Description	Related commands	Section Reference
Port Virtualizer (S-comp or Port Extender)	PV	16	Defines an S-comp or a port extender. Together with a PV, create a set of S-channels elements	Add Port Virtualizer Update Port Virtualizer Parameters Get Port Virtualizer Parameters	7.4.4.2
S-channel	SC	N/A	An S-channel created as part of an S-comp/M-comp element. There is no separate S-channel element. An S-channel connected directly to an external element (PF/VF/EMP) is considered as a VSI. An S-channel connected to a VEB or a VEPA is considered as part of the VEB/VEPA.	N/A	7.4.4.2
L2 Filter	L2	N/A	An L2 filter element. Such an element is created by default to connect a MAC and the first PF tied to this MAC. An L2 is not created as part of the topology. Rather it is implemented using the filtering options to the VSI tied to it.	N/A	7.4.4.3
VEB/Port Aggregator	VEB/VEPA	17	A VEB or a virtual port aggregator.	Add VEB Get VEB Parameters	
Virtual Station Interface	VSI	19	A VSI can be part of a VEB, VEPA or PV element and is connected to a host external element (PF, VF or EMP).	Add VSI Update VSI parameters Get VSI Parameters Delete Element	7.4.5.2

The SEIDs are allocated according to firmware convenience and a fixed mapping should not be assumed.

Table 7-49 lists the switching hierarchy and should be used by embedded firmware and host software as a common database.

Table 7-49. Switching Structure Representation

Entry in Table	Description	Notes
SEID	The handle to the element.	
Element Type	As listed in Table 7-47 and Table 7-48.	
Up Link Connection(s)	Indicates the SEID of the switching element connected directly below the current element. Below means towards the network in this case.	For sub-elements of a composed switch element, the uplink is the containing element.
Down Link Connection(s)	Indicates the SEID of the switching element connected directly above the current element. Above means towards the host in this case.	The down link is provided only for elements connected to PFs, VFs or to the EMP. In other cases, the value is zero.
Control Port	For elements with a control port. Points to the control port.	
Scheduler Node	The scheduler node associated with this switch element.	

Table 7-50 lists the representation of a switching configuration as shown in Figure 7-50.

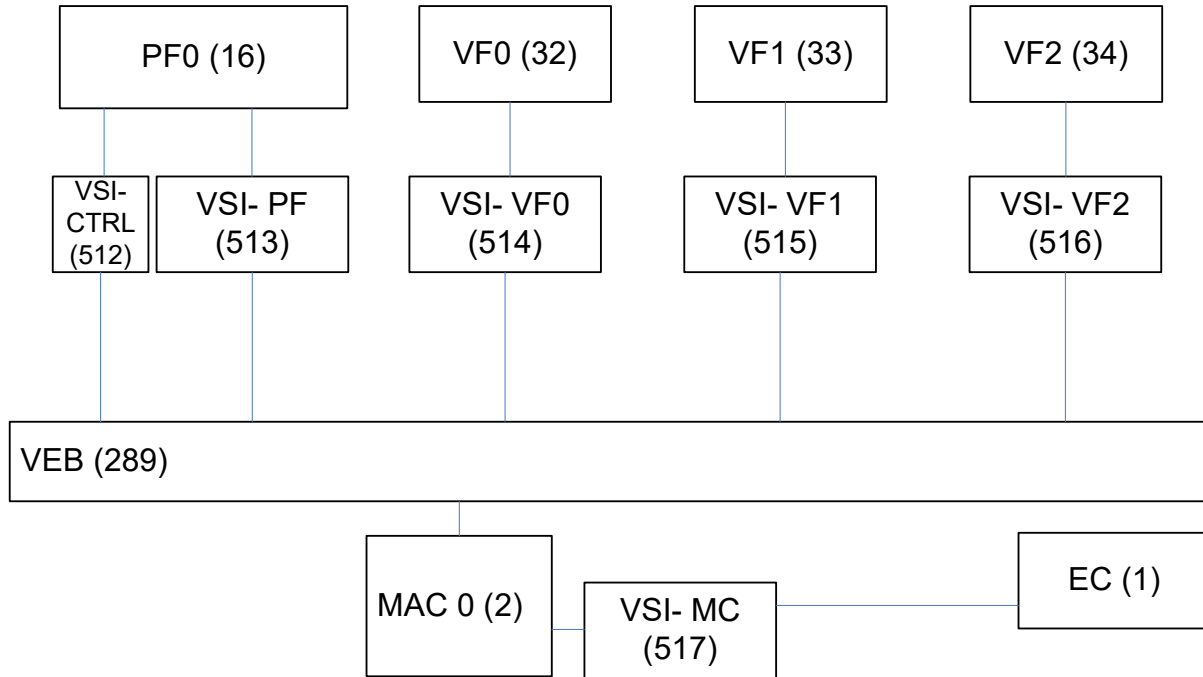


Figure 7-50. Switching Structure Example

Table 7-50. Switching Structure Representation Examples

SEID	Description	Element Type	Up Link	Down Link
0	Null	Null	0	0
1	Embedded controller	EMP	517	0
2	Port 0 MAC port	MAC	0	0
16	PF 0	PF	512	0
32	VF0	VF	514	0
33	VF1	VF	515	0
34	VF2	VF	516	0
289	VEB	VEB	2	0
512	VSI (PF)	VSI	289	16
513	VSI (control VSI for VEB)	VSI	289	16
514	VSI (VF0)	VSI	289	32
515	VSI (VF1)	VSI	289	33
516	VSI (VF2)	VSI	289	34
517	VSI (EMP)	VSI	2	1



7.4.9.2 Supported Switch Profiles

This section describes the supported per-port switching profiles.

The following profiles are supported by the X710-TM4:

1. Basic L2 (default configuration)
2. VEB/VEPA only
3. Port Virtualizer
4. Port Virtualizer and VEB/VEPA

Note: There is no central location in which a profile is set. These profiles are examples of possible configurations that are expected in the X710-TM4. The configuration of each profile is done using the admin commands described in [Section 7.4.9.4.2](#).

7.4.9.2.1 Basic L2

A basic L2 configuration is used for monolithic operating systems (non virtualized) where the network is not distributed to virtual channels. In this case, the switch configuration is as follows:

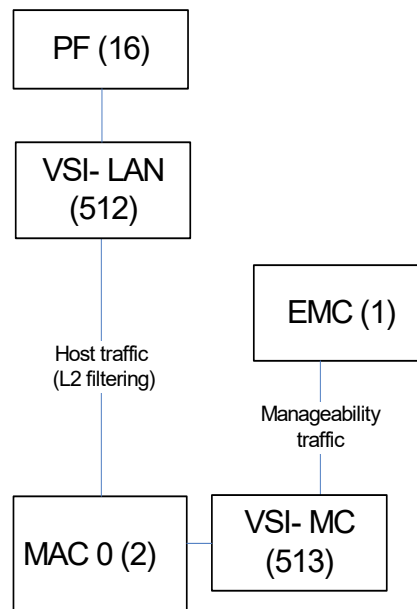


Figure 7-51. Switch Configuration - Basic L2

7.4.9.2.1.1 Basic L2 - Resource Allocation

In this profile, all resources allocated in the NVM to the port are available to this single function.

7.4.9.2.2 VEB/VEPA

A VEB configuration is used for virtualized operating systems where the network is not distributed to virtual channels. The switch configuration for the VEB/VEPA case is shown in Figure 7-52.

The use cases for VEB and VEPA are described in Section 7.4.2.1.2 and Section 7.4.2.2.1, respectively.

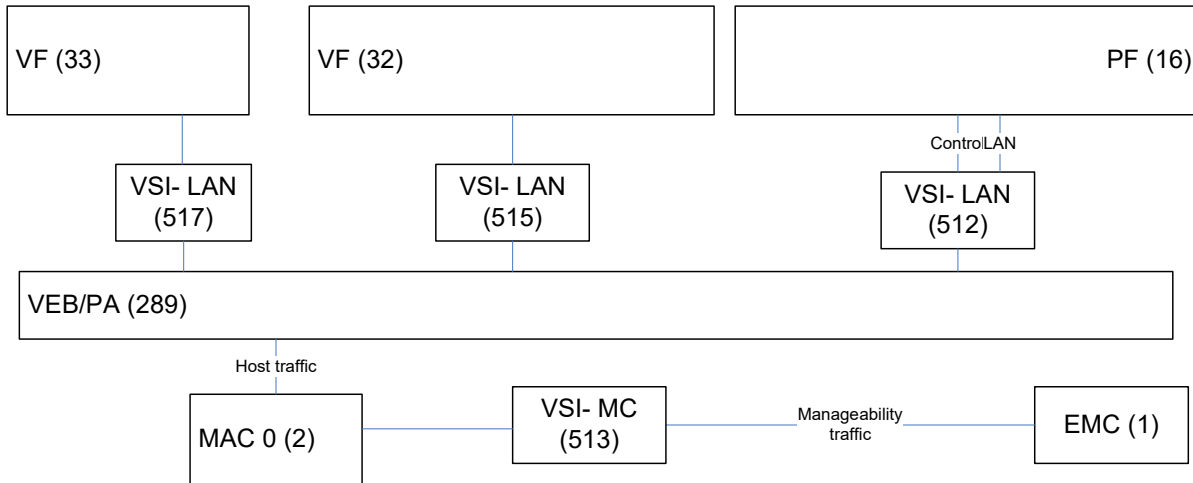


Figure 7-52. VEB/VEPA

Note: The control port of the VEB can be either a separate VSI or part of the regular VSI of the PF.

7.4.9.2.2.1 VEB - Resource Allocation

There is no limitation on the number VSIs on a single VEB (apart from the total number of VSIs supported) and up to 4 VEBs connected to a single PF.

The switching resources are allocated to the PF. The PF can decide to allocate the resources to any of the VEBs or to any of the VSIs according to its own policy.

7.4.9.2.3 Port Virtualizer

This mode is used when all the switching decisions are done in an external switch and each VSI is connected directly to one or more virtual channels. The usage models of the port virtualizer profile are described in Section 7.4.2.2.2. The switch configuration for the port virtualizer case is shown in Figure 7-53. In the case of a VF requiring multiple VSIs, it might request a new VSI using the Add VSI command. This command is forwarded to the PF for completion.

Note: The control port of the port virtualizer can be either a separate VSI or part of the regular VSI of the PF. In this case, the tagging of packets is controlled by the host. Packets sent from the control port should be non tagged.

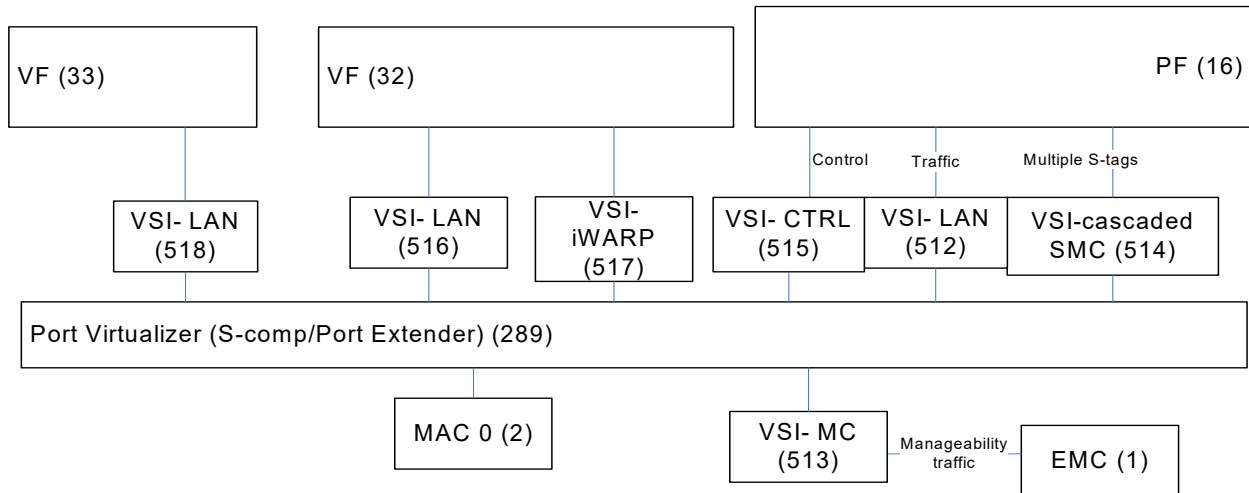
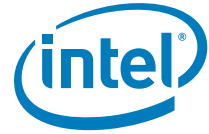


Figure 7-53. Port Virtualizer

7.4.9.2.4 Port Virtualizer and VEB/VEPA

A port virtualizer and VEB/VEPA configuration is used for virtualized operating systems where the network is distributed to virtual channels. The switch configuration for the port virtualizer and VEB/VEPA case is shown in Figure 7-54. In the case of a VF requiring multiple VSIs, it might request a new VSI using the Add VSI command. This command is forwarded to the PF for completion.

Note: The control port of the port virtualizer in this mode is the EMP, as there are multiple PFs connected to the same port virtualizer.

A special case of this mode is the mode where the link is divided into virtual channels and each channel is connected to a PF, but none of these are virtualized. In this case, there are no VEB/PAs in the device.

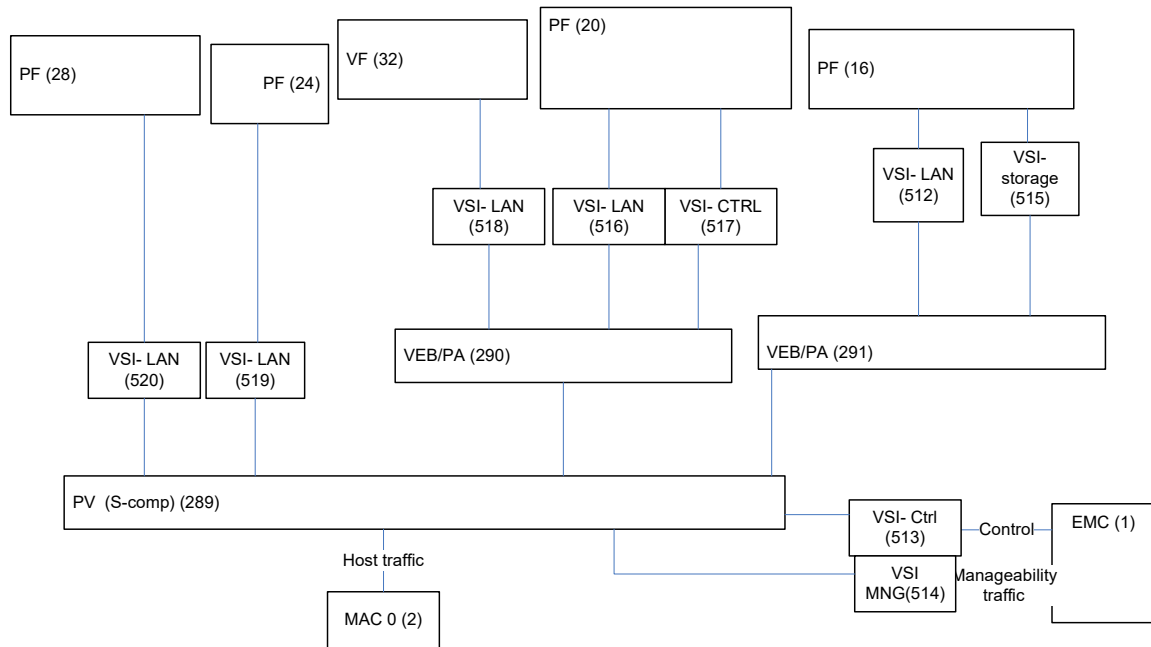


Figure 7-54. Port Virtualizer and VEB/VEPA

7.4.9.2.5 LAN/SAN S-comp

See Section 7.4.2.1.2 and Section 7.4.4.2.3 for more detail.

The initial configuration as exposed by the device is shown in Figure 7-55.

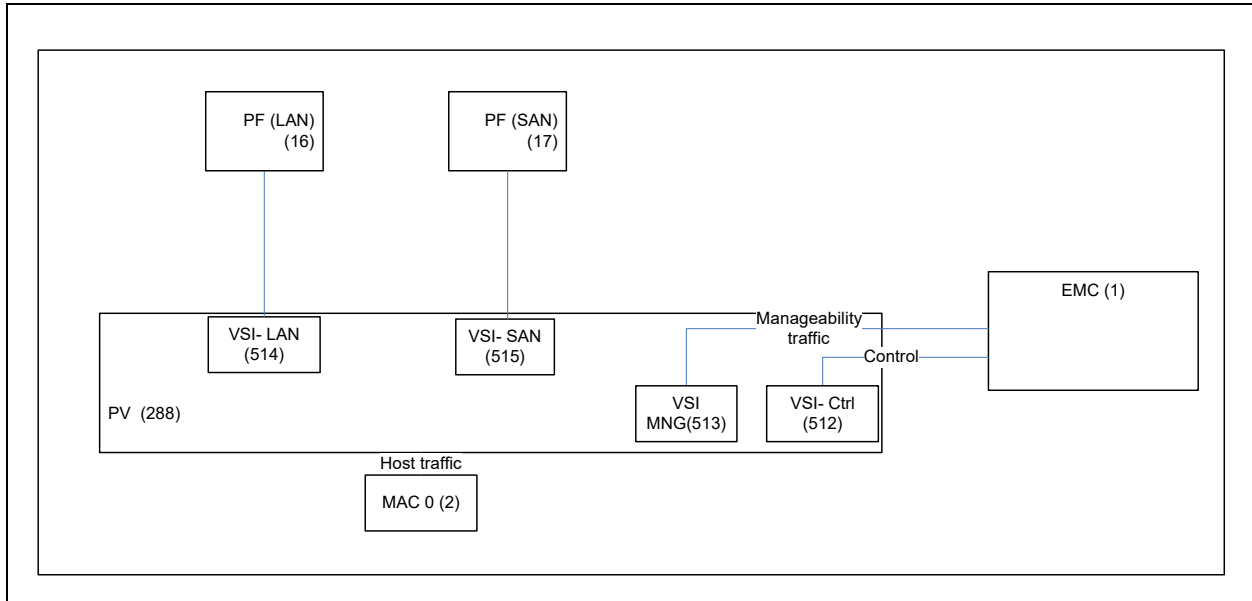


Figure 7-55. LAN/SAN Initial Configuration

7.4.9.3 Switch Resource Allocation

7.4.9.3.1 Allocated Resources

The tables that follow list the resources that are part of the switch and should be allocated to the different switch elements.

Table 7-51 lists the switching element resources. These resources are available no matter what the configuration of the forwarding tables is. Table 7-52 lists the forwarding rules resources. These resources are different for different configurations of the forwarding tables.

Note: Each element is associated with a port so two ports using the same value consumes two entries (such as the same MAC address).

Table 7-51. Switching Element Resources

Resource	Total Available	Notes
VEBs	16	There is no limitation on the number of VEBs assigned to a PF.
VSI	384	Up to 12 of the 384 VSIs are reserved for EMP traffic (LLDP and pass through traffic forwarding per port) and should not be allocated. VSIs are allocated from a shared pool by firmware and are not statically assigned.
VSI List Entries	6144	3 VSIs in each entry. These entries are used to create up to 2048 VSI lists such as multicast VSI lists or VLAN membership lists. VSI lists are allocated from a shared pool by firmware and are not statically assigned.
VLAN Statistic Pools	128	See Section 7.11.4.2 for details.



Table 7-51. Switching Element Resources

Resource	Total Available	Notes
TC Statistic Pools	128	Statically associated with the 16 VEBs (8 sets for 8 TCs for each of the VEBs).
Mirror Rules	64	
Queue Sets	1024	
Tunneling UDP Ports	16	

Table 7-52 lists the resources allocated according to the filter configurations used. The image used is defined in the Features enable.Switching mode NVM field.

Table 7-52. Switching Rules Resources

Resource (First-level Filters)	Size	Notes
Perfect Match MAC Addresses	1535	Each MAC address can be used for multiple VEBs within the same port. There can be up to 2048 VEB,MAC associations. One MAC address is statically allocated to the EMP.
VLANs	256	The entire range of 4096 VIDs can be covered; however, only 256 different VLANs can be stored in the forwarding tables. Each VLAN can be used in multiple VEBs within a port and there can be 1024 VEB,VLAN associations.
Ethertype Filters	64	These filters are tuples of the switch ID, Etherypes and direction.
S-tags	448	The entire range of 4096 S-VIDs can be covered; however, only 448 different S-tags can be stored in the forwarding tables.
Promiscuous Rules	1024	Unicast, broadcast or multicast promiscuous for different VEBs.
VLAN Promiscuous		The entire range of 4096 VIDs can be covered; however, only 1024 different VLANs/packet types (unicast, multicast, broadcast)/VEB can be stored in the forwarding tables.
Inner VLAN	512	Inner VLAN used for UDP cloud filtering.
VN Key Low / GRE Key/Tenant ID		
Inner MAC	1024	Each filter added occupies an entry in the table shared with the S-tag, VLAN mirroring filters.
Stag, VLAN Pairs (Mirroring)		Each filter added occupies an entry in the table shared with the inner MAC filters.
Destination IPv4 / IPv6 Address (Application IP)	128	Each filter added occupies an entry in the table shared with the inner MAC and inner VLAN filters.
MAC,VLAN Pairs	3072	For each entry in this table, an entry in the perfect match MAC addresses and a VLAN is used. The same is true for all the other types of filters.
MAC-Ethertype Filters	1024	Each entry is associated with an S-tag or switch ID. For each entry in this table, an entry in the perfect match MAC addresses and a Etherype filters is used. Note: An entry in the Mac Etherype table is taken for each direction of the rule (Tx or Rx).
MAC L2Filtering	512	This filter shares the same entries as the perfect match MAC address.
Inner MAC, Tenant ID	1024	
Outer MAC, Tenant ID, Inner MAC		
Inner MAC, Inner VLAN, Tenant ID	1024 ¹	For each entry in this table, an entry in the inner MAC addresses, inner VLAN and tenant ID filters are used.
Inner MAC, Inner VLAN Pair	1024 ²	For each entry in this table, an entry in the inner MAC addresses and inner VLAN filters entry are used.

1. Shared between Inner MAC, Inner VLAN, Tenant ID filters and Inner MAC filters.
2. Shared between Inner MAC, Inner VLAN filters and inner IP filters.



7.4.9.3.2 PFs Allocation Method

Resources can be categorized as shared resources that are used by multiple PFs and resources that are allocated to a single PF. Resources used by multiple PFs are inherently shared and cannot be allocated to any PF. Each PF requesting a resource gets it given there are still empty entries in the shared pool. It is the responsibility of each software device driver to release resources it doesn't use.

For resources allocated to a single PF, there is a basic allocation in the NVM for each PF (that can be zero). The resources not reserved to a PF are allocated on a first come first served basis.

The PF allocations structure in the NVM contains the resource allocation initial values.

If, after the enumeration stage, a PF is not enabled, its resources are given back to the shared pool.

Resources are allocated first from the dedicated pools and only if the dedicated pool of a PF is exhausted, a resource from the shared pool is given.

The resources that can be dedicated to a PF are:

- VEBs (TCs statistics pools are also allocated as part of the VEB).
- VSIs
- Perfect match MAC addresses
- S-tags
- VLAN Statistic pools
- Mirror rules
- Queue sets

All the other resources are considered as shared.

Note: The response of commands used to allocate dedicated resources includes information about the dedicated and shared resource left after the command is applied. The response of commands used to allocate shared resources includes information about the shared resource left after the command is applied.

The Get Switch Resources Allocation command returns the reserved allocation of each type of resource and the current level of usage of each resource.

7.4.9.3.2.1 Statistics Allocation

There are a few sets of statistics used by the switch as described in [Section 7.11.5](#). The allocation of statistics resources to the different switching elements is as follows:

- For each VSI, a set of per VSI statistics is allocated. The set to use is returned in the Add VSI response buffer in the *Statistic Counters* field. See [Section 7.11.4.3](#) for details.
- For each port and port virtualizer, a set of per port/port virtualizer statistics accessed using per-port registers is allocated as described in [Section 7.11.4.3](#).
- For each VEB allocated to a function, a set of VEB uplink statistics and per-VEB, per-TC statistics might be allocated as described in [Section 7.11.4.3](#), if statistics gathering is not disabled in the Add VEB command. The set to use is returned in the response of the Add VEB command in the *Statistic Counter* field.

In addition, a VEB can be allocated a set of statistics per-VLAN, per-VEB using the Add Statistics admin command. The set to use is returned in the *Statistic Counters* field in response of this command. Note that this request might fail if there are no free sets.



When statistics are not gathered, the following statistics are not valid for this VEB:

- GLPRT_RUPP[0]
- GLSW_GOTCH/L
- GLVEBVL_GOTC_[n]
- GLVEB_TCBCH/L[n]
- GLVEB_TCPCH/L[n]
- GLSW_UPTCH/L[n]
- GLSW_MPTCH/L[n]
- GLSW_BPTCH/L[n]
- GLSW_GORCH/L[n]
- GLVEB_VLBCH/L[n]
- GLVEB_RCBCH/L[n]
- GLVEB_RCPCH[n]
- GLSW_UPRCH/L[n]
- GLSW_MPRCH/L[n]
- GLSW_BPRCH/L[n]
- GLSW_RUPP_[n]
- GLVEB_VLUPCH/L[n]
- GLVEB_VLMPCH/L[n]
- GLVEB_VLBPCH/L[n]

7.4.9.3 VFs Allocation Method

The X710-TM4 manages VF resources as part of the PF resources. The PF driver should manage the VF requests based on the PF allocation.

7.4.9.4 Switch Initialization Flow

The initialization flow of the switch is composed as follows:

1. Pre BIOS stages:
 - a. Initialize the switch hardware ([Section 7.4.9.4.1](#)).
 - b. Initialize a basic switch configuration according to the NVM ([Section 7.4.9.4.2](#)).
 - c. Define the supported virtualization modes.
 - d. Update from external configuration source like EVB management protocol.
2. BIOS configuration ([Section 7.4.9.4.3](#)).
 - a. Update of the switch configuration according to SMASH CLP.
 - b. Configure the switch according to the updated configuration.
3. Software configuration ([Section 7.4.9.4.4](#)).
 - a. Optionally, create VEB or VEPA elements by a VMM.
 - b. Allocate MAC addresses and VLANs to VSIs.

At the end of stage 1, basic connectivity to the pass through management traffic and the EMP is enabled. In addition, WoL functionality is also available. If already known, the connectivity to PFs is also provided.



At the end of stage 2, connectivity to the host is added. This is the configuration exposed to the pre-boot software device driver and to the operating system.

7.4.9.4.1 Switch Hardware Initialization

The switch is implemented using a programmable logic, enabling parsing of different packet types and implementation of different forwarding rules. This programmable hardware is configured as part of the X710-TM4's auto-load of the Core Registers Auto-load NVM section. The lookup tables and the switching rules logic are configured at this stage.

This stage is part of the auto-load 2 initialization stage described in [Section 4.2.1.3](#).

This stage is activated at each core reset.

7.4.9.4.2 Initial Switch Configuration

The internal EMP firmware defines the initial switch structure. This stage is done after all hardware auto-load completes and the number of currently enabled PCIe PF functions is known. Before any NVM configuration, the following elements are defined by firmware:

Table 7-53. Initial Elements

Elements	SEID	Note
EMP	1	
Ports	2-5	Only for the ports enabled.
PFs	16-31	Only for the functions enabled.
VFs	32-159	For all VFs. VFs not used are not connected to VSIs.

At this stage, firmware loads the switch configuration from the NVM. The EMP settings NVM module contains the entire configuration needed. It also contains the following information relevant to the switch operation:

- A list of capabilities supported by the switch (type of filtering, types of connections, etc.).
- PF allocations module ([Section 7.4.9.4.2.2](#)).

The EMP settings module is described in [Section 6.2.1.14](#).

This stage is part of the firmware initialization stage described in [Section 4.2.1.4](#).

This stage is activated at each core reset.

7.4.9.4.2.1 Initial VSIs

At initialization, only minimal forwarding capabilities are needed:

- Forwarding of pass through traffic to the manageability interface.
- Forwarding of LLDP traffic to the EMP.
- WoL detection.

The initial switching configuration includes only the forwarding rules needed to support these flows. At this stage, the host is not connected to the network and cannot get traffic. If an additional pre-boot configuration is not expected, then the connectivity to the host is also done at this stage.



The basic configuration for port connectivity is achieved by connecting a single VSI to each port to one PF. The connection is from the port to the function connected to it with the lowest function number. The mapping of functions to the port is reflected in the PFGEN_PORTNUM.PORT_NUM field. These VSIs are created with a regular data port connection type and a PF VSI type.

All the VSIs created at this stage have the default switch ID of the port. The following parameters should be set to a non default value in these VSIs:

- Allow destination override should be set.
- For the PF VSIs, queue mapping should be set to contiguous and all the queues of the PF should be allocated to this VSI. The first queue is always queue 0 (of the PF).

As this flow is applied on each core reset, it is possible that a valid alternate RAM already exists. In this case, the alternate RAM-based configuration, as described in [Section 7.4.9.4.3](#), is applied.

7.4.9.4.2.2 PF Allocations

For each PF, the PF allocation sub-module in the EMP Core module contains guaranteed allocations for the following resources:

- VEBs (TCs statistics pools are also allocated as part of the VEB).
- VSIs
- Perfect match MAC addresses
- S-tags
- VLAN statistic pools
- Mirror rules queue sets
- Inner MAC
- IP addresses

After the basic topology is defined, an L2 filter should be added for each PF using the factory MAC address of this PF as stored in the PF allocations structure in the NVM or any alternate RAM override of those addresses. The *Load PF MAC Address* field in the PF Flags word in the PF allocation sub-module is used to define which MAC addresses to add to the switching decision. If this bit is set, only packets that passes the MAC and if needed, the S-tag are forwarded to the PF.

In addition, an L2 filter forwarding packets to the EMP should be added for each port using the port MAC addresses stored in the PRTGL_SAL and PRTGL_SAH registers.

By default, resources are allocated from the dedicated pool first and only when the dedicated pool is consumed or a *Use Shared* flag is set, the shared pool is used. A resource assigned from the dedicated pool can be used by a single PF only. An attempt to use it by another PF is treated as an error.

Firmware tracks and reports only the dedicated resources used by a function and not the shared ones.

Software can indicate a MAC address should be allocated from the shared pool, in which case it can be used by multiple PFs. This indication should be set for any resource expected to be used by multiple PFs (such as multicast addresses). MAC addresses assigned/removed as part of the Add/Remove Control Filter Aqs are taken from the shared pool.

VSIs are always allocated from the dedicated pool first and from the shared pool after that. Release of VSIs is from shared first and then from dedicated.

In order to use a resource from the shared pool, the *Use Shared* <resource name> flag should be set in the Add MAC, VLAN pair and in Add Cloud Filter Admin commands. Such a resource might be used by a multiple PFs. If not set, it is expected the resource is used by a single PF.



Note: Teamed ports using the same MAC address might allocate the MAC address from the dedicated pool, as MAC addresses are qualified with the port number.

7.4.9.4.3 BIOS Configuration

As part of the SMASH CLP stage, commands affecting the switch configuration might be received. These commands should be translated by the SMASH CLP code to the updates of the alternate RAM.

After all the SMASH CLP commands are received, the CLP code should update firmware with the content of the alternate RAM as described in [Section 4.3.2](#).

7.4.9.4.3.1 SFP Mode

For each of the enabled functions, a VSI is added and is connected to the MAC.

The VSIs created at this stage have the default switch ID of the port. The following parameters should be set to a non-default value in these VSIs:

- Allow destination override should be set.
- Queue mapping should be set to contiguous and all the queues of the PF should be allocated to this VSI. The first queue is always queue 0 (of the PF).

Once the alternate RAM is updated and executed by firmware, it might, depending on the configuration, disable some of the admin commands software is allowed to execute.

7.4.9.4.4 Operating System initialization

After operating system boot, each PF is connected to a physical port or an S-channel. Before adding further switching elements, the PF driver should use the Get Switch Configuration admin command ([Section 7.4.9.5.3.1](#)) to get the SEID of the switch element to which this function is connected.

After that, it might add VEB, VEPA or port virtualizer elements according to the instructions received from the switch control plane in the operating system using the admin commands described in [Section 7.4.9.5](#).

7.4.9.5 Programming Interface

The programming of the different switching elements is done using admin commands. Commands to configure a switching element can be received only from the control port of the element.

7.4.9.5.1 Switch Configuration Admin Commands Summary

[Table 7-54](#) lists the different commands used to configure switch elements.



Table 7-54. Switch Configuration Admin Commands (0x02xx)

Command	Opcode	Description	Reference
Generic Commands (0x020x)			
Get Switch Configuration	0x0200	Describe the networking structure of the port.	7.4.9.5.3.1
Add Statistics	0x0201	Add a statistics block to a VLAN in a switch.	7.4.9.5.3.2
Remove Statistics	0x0202	Remove a statistics block for a VLAN in a switch.	7.4.9.5.3.3
Set Port Parameters	0x0203	Defines the default parameters of a LAN port.	7.4.9.5.3.4
Get Switch Resources Allocation	0x0204	Reports the resources allocated to the PF.	7.4.9.5.3.5
Set Switch Configuration	0x0205	Configure switch global settings.	7.4.9.5.3.5
VSI Commands (0x021x)			
Add VSI	0x0210	Add a VSI to a switching element.	7.4.9.5.5.1
Update VSI	0x0211	Update parameters of a VSI.	7.4.9.5.5.2
Get VSI Parameters	0x0212	Get the parameters of a VSI.	7.4.9.5.5.3
Port Virtualizer Control (0x022x)			
Add Port Virtualizer	0x0220	Create an S-comp or a port extender.	7.4.9.5.6.1
Update Port Virtualizer Parameters	0x0221		7.4.9.5.6.2
Get Port Virtualizer Parameters	0x0222		7.4.9.5.6.3
VEB/Port Aggregator Control (0x023x)			
Add VEB	0x0230	Create a VEB/port aggregator.	7.4.9.5.7.1
Get VEB Parameters	0x0232	Get the parameters of a VEB/VEPA.	7.4.9.5.7.2
Switch Connectivity Configuration (0x024x)			
Delete Element	0x0243		7.4.9.5.8.1
Forwarding Table Configuration (0x025x)			
Add MAC,VLAN Pair	0x0250	Add a MAC/VLAN pair to the lookup table.	7.4.9.5.9.1
Remove MAC,VLAN Pair	0x0251	Remove a MAC/VLAN pair from the lookup table.	7.4.9.5.9.2
Add VLAN	0x0252	Add a VLAN to the VEB/port aggregator (can be regular or local, primary or secondary).	7.4.9.5.9.3
Remove VLAN	0x0253	Remove a VLAN or members of a VLAN from the VEB/port aggregator.	7.4.9.5.9.4
Set VSI Promiscuous Modes	0x0254		7.4.9.5.9.5
Add S-tag	0x0255	Add an of S-tags to a VSI.	7.4.9.5.9.6
Remove S-tag	0x0256	Remove an S-tag from a VSI.	7.4.9.5.9.7
Update S-tag	0x0259	Update the default S-tag of a VSI.	7.4.9.5.9.8
Add Control Packet Filter	0x025A	Add Ethertype (+MAC) filter to control port.	7.4.9.5.9.9
Remove Control Packet Filter	0x025B	Remove Ethertype (+MAC) filter from control port.	7.4.9.5.9.10
Add Cloud Filters	0x025C	Add a set of filters for cloud connections.	7.4.9.5.9.11
Remove Cloud Filters	0x025D	Remove a set of filters for cloud connections.	7.4.9.5.9.12
Clear all WoL Switch Filters	0x025E	Remove all WoL related filters.	7.4.9.5.9.13

**Table 7-54. Switch Configuration Admin Commands (0x02xx)**

Command	Opcode	Description	Reference
Replace Cloud Filters	0x025F	Re-define the use of the cloud filter switch tables.	7.4.9.5.9.13
Mirroring Configuration (0x026x)			
Add Mirror Rule	0x0260	Define a mirroring rule.	7.4.9.5.10.1
Delete Mirror Rule	0x0261	Remove a mirroring rule.	7.4.9.5.10.2

7.4.9.5.2 Configuration Flow Examples

This section describes examples of software configuration flows for different types of switch topologies.

The following examples are provided:

- An SFP with a port virtualizer used to distribute VM traffic. In this mode, each VF/VM is assigned a different S-channel.
- An SFP with a VEB used to distribute VM traffic. In this mode, each VF/VM is assigned a VSI within the VEB. The same flow can be used to add a VEPA element.
- An MFP with a port virtualizer used to distribute the physical ports traffic. In this mode, each PF is assigned a different S-channel.

More complex network topologies are supported. The configuration of such topologies can be inferred from the examples provided in the sections that follow.

For each example, the admin command to use and the main parameters in each command are described.

7.4.9.5.2.1 Port Virtualizer (SFP)

In this mode, internal firmware initiates a single VSI per port. This VSI is connected to the PF. Other VSIs might be connected to the EMP for MC or local traffic.

The software device driver should use the following commands to create a port virtualizer and connect a set of VSIs to it:

- Get Switch Configuration — This command provides the default VSI connected to the port.
- Get VSI — This command provides the content of the current VSI context. Note that the PF might update this context using an Update VSI command.
- Add Port Virtualizer — This command adds a port virtualizer on top of the port. The uplink SEID should be the physical port. The Default VSI SEID should be the default VSI received in the previous command.
- If needed, a separate control port might be added by using an Add VSI with the port virtualizer as the uplink SEID and a control port connection type.
- For each of the VMs/VFs that needs to be connected to the port virtualizer, an Add VSI should be sent with the port virtualizer as the uplink SEID, a regular data port connection type and a switch ID value equal to the S-tag assigned to this VM/VF. For VMs, the VSI type should be VM.

7.4.9.5.2.2 VEB (SFP)

In this mode, internal firmware initiates a single VSI per port. This VSI is connected to the PF. Other VSIs might be connected to the EMP for MC or local traffic.



The software device driver should use the following command to create a VEB and connect a set of VSIs to it:

- Get Switch Configuration — This command provides the default VSI connected to the port.
- Get VSI — This command provides the content of the current VSI context. Note that the PF might update this context using an Update VSI command.
- Add VEB — This command adds a VEB on top of the port. The uplink SEID should be the physical port. The default VSI SEID should be the default VSI received in the previous command. The software device driver should register the switch ID assigned to this VEB.
- If needed, a separate control port might be added by using an Add VSI with the VEB as the uplink SEID and a control port connection type.
- For each of the VMs/VFs that needs to be connected to the VEB, an Add VSI should be sent with the port virtualizer as the uplink SEID, a regular data port connection type and a switch ID value equal to the switch ID of the VEB. For VMs, the VSI type should be VM.
- For each VM/VF, Add MAC, VLAN pair commands should be sent to enable adequate forwarding of the traffic.

7.4.9.5.2.3 Port Virtualizer (MFP)

In this mode, internal firmware initiates a single port virtualizer and a VSI for each PF on the port. Other VSIs might be connected to the EMP for MC or local traffic.

The software device driver should use the following command to create a VEB and connect a set of VSIs to it:

- Get Switch Configuration — This command provides the default VSI connected to the port.
- Get VSI — This command provides the content of the current VSI context. Note that each PF might update this context using an Update VSI command.

7.4.9.5.3 Generic Commands (Opcode 0x020x)

7.4.9.5.3.1 Get Switch Configuration (0x0200)

This command is used to discover the switch configuration of the port. The software device driver must use this command as part of the initialization flow before adding new switching elements or manipulating existing ones.

This function is available to any PF. In an MFP case, each PF only sees the switch configuration owned by it. This usually includes the switching structure from the port virtualizer and upwards.

Table 7-55. Get Switch Configuration Command (Opcode: 0x0200)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0200	Command opcode.
Datalen	4-5		Length of response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.



Table 7-55. Get Switch Configuration Command (Opcode: 0x0200)

Name	Bytes.Bits	Value	Remarks
First SEID	16-17	0x0	If not zero, start the report from the requested SEID (used for continuation commands).
Reserved	18-23	0x0	Must be zero.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		

Table 7-56 lists the response buffer for the Get Switch Configuration command.

Table 7-56. Get Switch Configuration Response (Opcode: 0x0200)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0200	Command opcode.
Datalen	4-5		Length of response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Next SEID	16-17	0x0	If not zero, indicates that not all the configuration was returned and a new command should be sent with this value in the First SEID field.
Reserved	18-23	0x0	Must be zero.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		

Table 7-57 lists the structure of the response buffer for the Get Switch Configuration command.

Table 7-57. Get Switch Configuration Command Response Buffer

Offset (Bytes)	Description
0-15	Header. See Table 7-58 for detail.
16-31	Switch Element #1. See Table 7-59 for detail.
...	
16*n+15-16*n	Switch Element #n. See Table 7-59 for detail.



Table 7-58. Get Switch Configuration Command Response Header

Offset (bytes)	Description
0-1	Number of switching elements in the structure. The size of the buffer is 16 * (number of elements + 1). If the buffer size is too small to return all the switching elements, only the elements fitting the buffer are returned and the software device driver might request the subsequent elements using a different first element. The maximum number of entries that can be returned in a single command is 255.
2-3	Total number of switching elements. The total number of switching elements. This might be larger than the number of elements in the structure.
4-15	Reserved.

Table 7-59. Get Switch Configuration (Switch Element)

Offset (Bytes)	Description	Notes										
0	Element type as listed in Table 7-47 and Table 7-48 .											
1	Revision. Describes the revision of the element type. For the X710-TM4, the revision is always 1.											
2-3	SEID. Defines the switch element ID of this element.											
4-5	Up Link Connection. Indicates the SEID of the switching element connected directly below the current element. Below means towards the network.	For sub-elements of a composed switch element, the uplink is the uplink of the entire switch element.										
6-7	Downlink Connection. Indicates the SEID of the switching element connected directly above the current element. Above means towards the host.	The down link of a composed element is the default port of this element.										
8-10	Reserved.	Reserved.										
11	Connection Type. 0x0 = Reserved. 0x1 = Regular data port. 0x2 = Default port. 0x3 = Cascaded port virtualizer port. 0x4 - 0xFF = Reserved.	Defines the type of connection to the uplink element.										
12 -13	Reserved											
14-15	Element Specific Information. Provides additional information according to the element type as follows: <table border="0"> <thead> <tr> <th>Element Type</th> <th>Information</th> </tr> </thead> <tbody> <tr> <td>MAC (1)</td> <td>The physical port number</td> </tr> <tr> <td>PF (2)</td> <td>The physical function number</td> </tr> <tr> <td>VF (3)</td> <td>The virtual function number</td> </tr> <tr> <td>VSI (19)</td> <td>The VSI number</td> </tr> </tbody> </table> This field is reserved for all other element types.	Element Type	Information	MAC (1)	The physical port number	PF (2)	The physical function number	VF (3)	The virtual function number	VSI (19)	The VSI number	
Element Type	Information											
MAC (1)	The physical port number											
PF (2)	The physical function number											
VF (3)	The virtual function number											
VSI (19)	The VSI number											

7.4.9.5.3.2 Add Statistics (0x0201)

The X710-TM4 supports 128 smonVlanStats counters as described in [Section 7.11.4.3](#). This command is used to allocate a set of smonVlanStats counters to a specific VLAN in a specific switch.

**Table 7-60. Add Statistics Command (Opcode: 0x0201)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0201	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Switch SEID	16-17		Defines the SEID of the switch for which the stats are requested.
VLAN ID	18-19		The VLAN ID for which the statistics are requested. 15:12 = Reserved. 11:0 = VLAN ID.
Statistic Counter	20-21	Statistic Counters Index	Zeroed by the software device driver. Firmware returns an index of the statistics counters block assigned to this VLAN.
Reserved	22-31	0x0	Reserved.

Table 7-61. Add Statistics Response (Opcode: 0x0201)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0201	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if SEID does not point to a valid switch. ENOSPC - if there aren't enough resources to assign a statistics block. EINVAL - a statistic block is already allocated to this VLAN on this VEB.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Switch SEID	16-17		Copied from command.
VLAN ID	18-19		Copied from command.
Statistic Counter	20-21	Statistic Counters Index	Zeroed by the software device driver. Firmware returns an index of the statistics counters block assigned to this VLAN.
Reserved	22-31	0x0	Reserved.

7.4.9.5.3.3 Remove Statistics (0x0202)

The X710-TM4 supports 128 smonVlanStats counters as described in [Section 7.11.4.3](#). This command is used to deallocate a set of smonVlanStats counters from a specific VLAN in a specific switch.

**Table 7-62. Remove Statistics Command (Opcode: 0x0202)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0202	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Switch SEID	16-17		Defines the SEID of the switch for which the stats are currently allocated.
VLAN ID	18-19		The VLAN ID for which the statistics are currently allocated. 15:12 = Reserved. 11:0 = VLAN ID.
Statistic Counter	20-21	Statistic Counters Index	The statistics counters block assigned to this VLAN.
Reserved	22-31	0x0	Reserved.

Table 7-63. Remove Statistics Response (Opcode: 0x0202)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0202	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if SEID does not point to a valid switch. EINVAL - if the statistic block mentioned is not pointed by this VLAN switch combination.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31		Reserved.

7.4.9.5.3.4 Set Port Parameters (0x0203)

This command is used to define the default parameters of a physical port. When received in MFP mode, the last configuration received is used. In MFP mode, the configuration is not reset by a PFR, only by a device wide reset. The only exception is pass bad frames which is reset, if the PF to which the bad frames VSI is associated is reset.



Table 7-64. Set Port Parameters Command (Opcode: 0x0203)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0203	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Command Flags	16-17	Bitfield	0 = Save Bad Packets. If set, packets with errors are forwarded to the bad frames VSI. 2 = Enable Double VLAN. If set, this port expects double VLAN packets. Should not be set if the channel identifier as reported in Discover Device/Function Capabilities Switching mode is set to VLAN. Remaining bits are reserved.
Bad Frames SEID	18-19		Defines the SEID of the VSI to which bad frames are forwarded. Bit 15 = Valid SEID - ignored in this field. Bit 14:10 = Reserved. Bit 9:0 = SEID number of the VSI. Note: Relevant only if Command Flags.Save Bad Packets is set.
Reserved	20-31	0x0	Reserved.

Table 7-65. Set Port Parameters Response (Opcode: 0x0203)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0203	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. The following response might be returned by this command: EPERM - if the operation is not permitted. For example, set the outer VLAN when the outer VLAN is not enabled.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Command Flags	16-17	Bitfield	
Bad Frames VSI	18-19		
Default Switch ID	20-21		Returns the Switch ID assigned to the port.
Reserved	22-31	0x0	Reserved.

7.4.9.5.3.5 Get Switch Resources Allocation (0x0204)

This command is used to query the resources allocated to a function.



Table 7-66. Get Switch Resources Allocation Command (Opcode: 0x0204)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0204	Command opcode.
Datalen	4-5		Length of response buffer. Should be big enough to contain the expected response buffer size.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-23	0x0	Reserved.
Data Address High	24-27		Return Buffer Address.
Data Address Low	28-31		

Table 7-67. Get Switch Resources Response (Opcode: 0x0204)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0204	Command opcode.
Datalen	4-5		Length of buffer.
Return Value/ VFID	6-7		Return value. There are no specific errors to this command.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Number of Entries	16	0xD/0x14 ¹	Number of resource entries.
Reserved	17-23	0x0	Reserved.
Data Address High	24-27		Return Buffer Address.
Data Address Low	28-31		

1. In cloud images, Features Enable.Switching mode is set to cloud or UDP cloud, then the relevant filters are also returned.

The return buffer contains structures of 16 bytes for each resource type of the following format. The number of entries is returned in the number of entries field in the response.



Name	Bytes.Bits	Description
Resource Type	0	0x0 = VEBs. 0x1 = VSIs. 0x2 = Perfect match MAC addresses. 0x3 = S-tags. 0x4 = Reserved. 0x5 = Reserved. 0x6 = Reserved. 0x7 = VLANs. 0x8= VSI list entries (3 VSI in each entry). 0x9 = Reserved. 0xA = VLAN Statistic pools. 0xB = Mirror rules. 0xC = Queue sets ¹ . 0xD = Inner VLAN forward filters. 0xE = Reserved. 0xF = Inner MACs. 0x10 = IPs. 0x11 = GRE/VN1 keys. 0x12 = VN2 keys. 0x13 = Tunneling ports. 0x14- 0xFF = Reserved.
Reserved	1	Reserved.
Guaranteed Allocation	2-3	Number of resources from this type guaranteed to this function. For example, the value set in NVM for this PF and this resource type described in Section 6.2.24 .
Total Dedicated	4-5	Total number of dedicated resources from this type available to all functions.
Currently Used Dedicated	6-7	Number of dedicated resources from this type used by this function.
Total Not Allocated	8-9	Total number of resources from this type still not allocated and not reserved by any function. The number of free shared resources currently available plus dedicated resources of the PF not allocated.
Reserved	10-15	Reserved.

1. When multi-queue set mode is enabled for a port, the total number of fields not allocated is zero for the resource type queue sets (0xC).

7.4.9.5.4 Set Switch Configuration Command

This command is used to set device-wide switch configurations. The last configuration of any software device driver is used and overrides previous settings.

Table 7-68 Set Switch Configuration command (Opcode: 0x0205)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0205	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.



Table 7-68 Set Switch Configuration command (Opcode: 0x0205)

Name	Bytes.Bits	Value	Remarks
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Flags	16-17	0x0	<p>0 = Promiscuous behavior:</p> <ul style="list-style-type: none"> 0 = Packets are forwarded according to promiscuous filter even if matching an exact match filter. 1 = Packets are forwarded according to promiscuous filter only if not matching an exact match filter. <p>1 = Enable L2 filtering. If set, the L2 filter table should pass in addition to the regular forwarding decision. 2 = Enable automatic ATR eviction. 15:2 = Reserved.</p>
Valid Flags	18-19	0x0	<p>0 = Promiscuous behavior valid. 1 = Enable L2 filtering valid. 15:2 = Reserved.</p>
Switch Tag	20-21		<p>Used for internal switching, in case S-Comp is not defined. The settings of the Switch Tag, First Tag, and Second Tag fields must not result in the Switch Tag value being identical to either the First Tag or Second Tag values.</p> <p>Note that packets (ingress or egress) containing this value as the Ethertype after MAC DA/SA will be dropped. If this field is 0, no change is made to the setting.</p>
First Tag	22-23		<p>If Double VLAN is used, this field specifies the Ethertype of the outer VLAN tag. If Double VLAN has not been enabled in the Set Port Parameters command, this field is ignored. If this field is 0, no change is made to the setting.</p>
Second Tag	24-25		<p>If double VLAN is used, this field specifies the Ethertype of the inner VLAN tag. If double VLAN has not been enabled in the Set Port Parameters command, this field specifies the Ethertype of the single VLAN tag. If this field is zero, no change is made to the setting.</p>
L4 Port Filter Mode	26		<p>Bit 7 = Valid. 0 = No action. 1 = Switch to the configuration defined by bits 6:0.</p> <p>Bit 6 = SRC/DST L4 port. 0 = Destination L4 port. 1 = Source L4 port.</p> <p>Bits 5:4 L4 type. 0 = Reserved 1 = TCP. 2 = UDP. 3 = Both TCP and UDP.</p> <p>Bits 3:0 mode. 0 = Default mode. 1 = L4 Port only mode. 2 = Non-tunneled mode. 3 = Tunneled mode. All other values are reserved.</p>
Reserved	27-31	0x0	Reserved.



Table 7-69 Set Switch Configuration Response (Opcode: 0x0205)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0205	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/VFID	6-7		Return value. EINVAL - Invalid parameters. EBUSY - The L4 port filter mode change requires filters that are in use. ENXIO - The L4 port filter mode change requires filters that have already been replaced.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31		Reserved.

7.4.9.5.5 VSI Commands (Opcode 0x021x)

7.4.9.5.5.1 Add VSI (0x0210)

This command is used to add a new VSI. A VSI must connect to an existing switching element. A function can connect a VSI only to a switching element it controls.

When an Add VSI command is received, internal firmware checks if all the requested resources are available and allocate them to the VSI. If part of the resources are not available, the command returns a list of unavailable resources.

Table 7-70. Add VSI Command (Opcode: 0x0210)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0210	Command opcode.
Datalen	4-5	0x80	Length of buffer.
Return Value/VFID	6-7	0x0	Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Uplink SEID	16-17	0x0	Defines the uplink SEID to which this VSI should be connected.
Connection Type	18	0x0	Defines to which port of the uplink element this VSI connects. 0x0 = Reserved. 0x1 = Regular data port. 0x2 = Default port. 0x3 - 0xFF = Reserved. Note: If set as the default port, replaces the existing default port of the underlying switching element.
Reserved	19	0x0	Reserved.



Table 7-70. Add VSI Command (Opcode: 0x0210)

Name	Bytes.Bits	Value	Remarks
VF Function Number	20		Defines the VF function to which this VSI connects. Valid only if Function Type is VF. Should be ignored if VSI type is not VF. Note: The VF number here is the absolute VF number (0-127) and not the number relative to the PF first VF.
Reserved	21		Reserved.
Command Flags	22-23		1:0 = VSI type: <ul style="list-style-type: none"> • 0 = VF. • 1 = VMDq2 (VM). • 2 = PF. • 3 = EMP/MNG. 2 = Cascaded Port Virtualizer. The S-tag manipulation commands can be used only if this bit is set. Bit 15:3 = Reserved.
Data Address High	24-27		Address of the buffer.
Data Address Low	28-31		

Table 7-71 lists the structure of the command buffer for the Add VSI command.

Table 7-71. Add VSI Command Buffer

Category	Byte/Bit	Field	Description
Valid Sections	0-1	Defines which sections are valid in the command.	0 = Switching section is valid. Must be set for Add VSI commands when the containing SEID is a Port Virtualizer. 1 = Security section is valid. 2 = VLAN handling section is valid. 3 = Cascaded port virtualizer section is valid. 4 = Ingress UP translation section is valid. 5 = Egress UP translation section is valid. 6 = Queue mapping section is valid. Must be set for Add VSI commands. If set for an Update VSI command, modified queues must be disabled. 7 = Queuing option section is valid. 8 = Outer UP translation section is valid. Should not be set if the inner UP to outer UP mapping is the identity mapping. 9 = Scheduler section is valid. This bit is ignored in the Update VSI command. 10-15 = Reserved.



Table 7-71. Add VSI Command Buffer

Category	Byte/ Bit	Field	Description
Switching	2-3.3	Switch ID	Defines the switch ID to which this VSI belongs. If not, S-tag is cleared and then this switch ID is an S-tag; otherwise, it is a virtual switch ID. If the VSI is connected to a port virtualizer, this value must be set. For a port connected to a VEB or directly to the MAC, this value and the Is Not S-tag field might be left at zero and the VEB switch ID or the default switch ID is used. An Update VSI command should be a valid tag (could also be zero for VEB).
	3.4	Is not S-tag	Refer to the Switch ID parameter for the usage of this bit.
	3.5	Allow Loopback	This bit should be set for VSIs that are connected to a VEB and should be cleared otherwise. A VSI connected to a VEB might have this bit cleared if used only for connecting to the link (like a management VSI). Cleared if section is not valid.
	3.6	Allow Local Loopback	This bit should be set for VSIs that are used as an uplink to a software (cascaded) VEB, VEPA or port virtualizer. Cleared if section is not valid.
	3.7-5.7	Reserved	Reserved for future switching parameters. Must be zero.
Security	6.0	Allow Destination Override	Allow the VSI to override the switching decision and fix the destination of a transmit packet. This bit should be set only for control ports. Cleared if section is not valid.
	6.1	Enable VLAN Anti Spoof	Cleared if section is not valid. Note: Enabling this mode may impact the transmit performance. It is recommended to use only the MAC anti spoof mode.
	6.2	Enable MAC Anti Spoof	Cleared if section not valid.
	6.3 - 7.7	Reserved	Reserved for future security parameters. Must be zero.



Table 7-71. Add VSI Command Buffer

Category	Byte/Bit	Field	Description
VLAN Handling	8-9	PVID + Default UP (16 Bits)	VLAN ID to use in Port based VLAN insertion. This field is relevant only if the <i>Insert PVID</i> field is set. Cleared if section not valid.
	10-11	Reserved	Reserved.
	12.0:1	VLAN Driver Insertion Mode	This field defines if the software device driver is allowed/should add a VLAN tag to the packets it sends. If the <i>Insert PVID</i> field is set, this field should be set to 01b. 00b = Reserved. 01b = Admit untagged/priority tagged only. Enable only packets without VLAN or with VLAN tag = 0. 10b = Admit.1Q tagged only. Enable only packets with VLAN. 11b = Enable all packets. If section is not valid, the default value is 11b.
	12.2	Insert PVID	Port-based VLAN Insertion. This bit controls the port-based insertion of VLANs. Should be set for VFs/VMDq2 VSIs according to the VMM request. If this field is set, the <i>PVID + Default UP</i> field should be set to the port-based VLAN. Cleared if section is not valid.
	12.3:4	VLAN and UP Expose Mode (Rx)	This field defines how received VLAN are handled. For non VF VSIs, 00b or 11b should be used. For VF VSIs, the mode should be set according to the VLAN awareness of the VM and the offload requested. 00b = Show VLAN, DEI and UP in descriptor (legacy behavior). 01b = Hide VLAN, DEI and show UP (VLAN ID = 0). 10b = Hide VLAN, DEI and UP. 11b = Do nothing (leave VLAN in packet). If section is not valid, the default value is 11b.
	12.5-15.7	Reserved	Reserved for future port VLAN parameters. Must be zero.
Ingress UP Translation	16-19	Ingress UP Translation Table	Defines the UP translation table for received packets according to the following list: 16.0:16.2 UP set if received UP is 0. 16.3:16.5 UP set if received UP is 1. 16.6:17.0 UP set if received UP is 2. 17.1:17.3 UP set if received UP is 3. 17.4:17.6 UP set if received UP is 4. 17.7:18.1 UP set if received UP is 5. 18.2:18.4 UP set if received UP is 6. 18.5:18.7 UP set if received UP is 7. 19: Reserved. This map is used to translate the 802.1P user priority bits received in the packet to the user priority exposed to the host. Relevant only if the UP is exposed to the host (VLAN and UP expose mode not equal 10b). If section is not valid, the default value is identity mapping.



Table 7-71. Add VSI Command Buffer

Category	Byte/Bit	Field	Description
Egress UP Translation	20-23	Egress UP Translation Table	<p>Defines the UP translation table for transmit packets according to the following list:</p> <ul style="list-style-type: none"> 20.0:20.2 UP set if sent UP is 0. 20.3:20.5 UP set if sent UP is 1. 20.6:21.0 UP set if sent UP is 2. 21.1:21.3 UP set if sent UP is 3. 21.4:21.6 UP set if sent UP is 4. 21.7:22.1 UP set if sent UP is 5. 22.2:22.4 UP set if sent UP is 6. 22.5:22.7 UP set if sent UP is 7. <p>23: Reserved.</p> <p>This map is used to translate the 802.1P user priority bits sent by the host to the user priority sent to the network.</p> <p>Note that the resulting user priority is further translated using the per TC translation table.</p> <p>If section is not valid, the default value is identity mapping.</p>
Cascaded Port Virtualizer	24-25	S-tag	<p>The S-tag (S-VID) to be inserted in transmit packets. If the Switch ID parameter is set and is an S-tag and not a cascaded port virtualizer, the first 13 bits of the S-tag should be set to the same value as the Switch ID parameter.</p> <p>This field is relevant only if the <i>S-tag Insert Enable</i> field is set.</p> <p>This field is 16 bits and should also include the default PCP priority bits to insert in the S-tag.</p> <p>Cleared in SFP mode and equal to the S-channel S-tag in MFP mode if section is not valid.</p>
	26.0:1	S-tag Extract Mode	<p>This field defines how received S-tags are handled.</p> <ul style="list-style-type: none"> 00b = Do nothing (cascaded port virtualizer without offload). 01b = Extract tag and do not insert in descriptor (regular VM). 10b = Extract tag from packet and expose in descriptor (cascaded port virtualizer with offload). 11b = Reserved. <p>If section is not valid, the default value is 00b in SFP mode and 01b in MFP mode.</p>
	26.2:3	Reserved	Reserved.
	26.4	S-tag Insert Enable	<p>This bit controls the port-based insertion of S-tags. Should be set if VSI is part of an S-channel and is not a cascaded port virtualizer VSI.</p> <p>If section is not valid, the default value is cleared in SFP mode and set in MFP mode.</p> <p>When this bit is set, the <i>Accept tag from host</i> bit should be cleared.</p>
	26.5	Reserved	Reserved.
	26.6	Accept Tag From Host	<p>Allow the host to insert an S-tag in a descriptor or in a packet. Should be set only for cascaded port virtualizer or control ports.</p> <p>Set in SFP mode and cleared in MFP mode if section is not valid.</p> <p>When this bit is set, the S-tag insert enable bit should be cleared.</p>
	26.7-27.7	Reserved	Reserved for future port S-tag parameters. Must be zero.



Table 7-71. Add VSI Command Buffer

Category	Byte/Bit	Field	Description
Queue Mapping	28.0	Mapping Method	Selects between a contiguous range of queues for this VSI versus scattered range: 0b = The VSI is assigned a contiguous range of PF queues. 1b = The VSI is assigned a scattered range of PF queues.
	28.1 - 29	Reserved	Reserved. Must be zero.
	30-61	Queue Mapping	If the mapping method = 0 (contiguous): 30.0:31.2 = The first queue allocated for this VSI in the PF space. This VSI can access all the queues from this queue to the end of the PF queues allocation, but is bound by the receive queues per TC configuration. 31.3:61 = Reserved. If mapping method = 1 (scattered): For each of the queues in the VSI, defines the actual queue in the PF according to the following encoding: For queue 'n' of the VSI offsets 30+2n - 31+2n are used to define the mapping to PF queues. For example, for queue 0: <ul style="list-style-type: none"> 30.0:31.2 = The PF queue matching allocated to queue 'n' of the VSI. For non-allocated queues, a value of 0x7FF should be set. 31.3:31.7 = Reserved. The same mapping is used for the next queues. In this method, up to 16 queues can be assigned. Note: For VSIs assigned to VFs, only the scattered method can be used. The first queue (in both modes) is the default queue of the VSI to which packets not queued by any filter is sent.
	62-77	Number and Offset of Queue Pairs per TCs	Fixes the number of queue pairs assigned to the VSI for each TC and the offset of these queues. 62.0 - 63.0 = Queue offset for TC0. 63.1 - 63.3 = Number of queues allocated to TC0. The actual number is 2^n. The allowed number of queues are: 1; 2; 4; 8; 16; 32; 64. 63.4 - 63.7 = Reserved. Note: If no queues need to be associated to a TC, the queue offset should be set to 0 and the number of queues to 0 (1 queue). This way, traffic associated with this TC is sent to the default queue. The following addresses are used with the same format for the next TCs: 64:65 = TC1. 66:67 = TC2. 68:69 = TC3. 70:71 = TC4. 72:73 = TC5. 74:75 = TC6. 76:77 = TC7.
Queueing Options	78.0:3	Reserved	Reserved.
	78.4	TCP Packets Enable	Reserved and cleared.
	78.5 - 81	Reserved	Reserved for future queueing parameters. Must be zero.
Scheduler	82	Enabled TCs	A bitmap indicating which TCs are enabled in this VSI. If section is not valid, only TC0 is enabled.
	83	Reserved	Reserved.



Table 7-71. Add VSI Command Buffer

Category	Byte/Bit	Field	Description
Outer UP Mapping	84-87	Egress Inner UP to Outer Translation Table	<p>Defines the UP translation table for transmit packets from inner to outer UP according to the following list:</p> <ul style="list-style-type: none"> 84.0:84.2 Outer UP set if inner UP is 0. 84.3:84.5 Outer UP set if inner UP is 1. 84.6:85.0 Outer UP set if inner UP is 2. 85.1:85.3 Outer UP set if inner UP is 3. 85.4:85.6 Outer UP set if inner UP is 4. 85.7:86.1 Outer UP set if inner UP is 5. 86.2:86.4 Outer UP set if inner UP is 6. 86.5:86.7 Outer UP set if inner UP is 7. <p>87: Reserved.</p> <p>This map is used to translate the 802.1P user priority bits on the inner UP to outer UP.</p> <p>If section is not valid, the default value is identity mapping. In order to get a fixed outer UP, all the values should be set to the requested outer UP.</p>
Reserved		Reserved	Reserved.
Response Space	96-127	Reserved	Reserved for response space.

Table 7-72 and Table 7-73 list the Add VSI response and the response buffer.

Table 7-72. Add VSI Response (Opcode: 0x0210)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0210	Command opcode.
Datalen	4-5	0x80	Length of buffer.
Return Value/ VFID	6-7		<p>Return value.</p> <p>The following error values can be returned:</p> <ul style="list-style-type: none"> ENOENT - if the uplink SEID or the function number do not point to valid elements. EINVAL - if the topology created is not valid. ENOSPC - if there aren't enough resources to assign a VSI.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID	16-17		<p>If the return value is zero, returns the SEID of the VSI.</p> <p>If the return value is ENOSPC, contains a bitmap that indicates which resources are missing:</p> <ul style="list-style-type: none"> 0 = No VSI left. 1 = Not enough scheduler nodes. 2 = Not enough statistics counters. 3 = Not enough switching entries. 4-15 = Reserved.
VSI Number	18-19		Returns the assigned VSI number.
VSI's Used	20-21		Number of VSI's used by this function.
Total VSI's Not Allocated	22-23		Total number of VSI's still not allocated and not reserved by any function.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		



Table 7-73. Add VSI Response Buffer

Category	Byte/Bit	Field	Description
Command Space	0-95	Reserved	Reserved for command space. Should contain the values provided by the software device driver.
Queue Set Handles	96-97	QS_Handle 0	The handle for queue set of TC0. Bits [9:0] of this handle are used by software to program the <i>RDYList</i> field in the transmit queues context for queues associated with TC0.
	98-99	QS_Handle 1	The handle for queue set of TC1. Same format as QS_Handle 0.
	100-101	QS_Handle 2	The handle for queue set of TC2. Same format as QS_Handle 0.
	102-103	QS_Handle 3	The handle for queue set of TC3. Same format as QS_Handle 0.
	104-105	QS_Handle 4	The handle for queue set of TC4. Same format as QS_Handle 0.
	106-107	QS_Handle 5	The handle for queue set of TC5. Same format as QS_Handle 0.
	108-109	QS_Handle 6	The handle for queue set of TC6. Same format as QS_Handle 0.
	110-111	QS_Handle 7	The handle for queue set of TC7. Same format as QS_Handle 0.
Statistic Counter	112-113	Statistic counters index	Returns an index of the statistics counters block assigned to this VSI.
Reserved	116-127	Reserved	Reserved.

7.4.9.5.5.1.1 Add VSI Settings Recommendations

Table 7-74 lists the recommended settings for common types of VSIs:

Table 7-74. Add VSI Recommended Settings

Category	Field	VSI Type						
		PF	VF	VMDq2	VMDq1	Cascaded VEB	Cascaded S-comp	Floating VEB
Valid Sections	Switching Section is Valid	0	1	1	1	1	1	1
	Security Section is Valid	1						
	VLAN Handling Section is Valid	1						
	Cascaded Port Virtualizer Section is Valid	1						0
	Ingress UP Translation Section is Valid	0						
	Egress UP Translation Section is Valid	0						
	Queue Mapping Section is Valid	1						



Table 7-74. Add VSI Recommended Settings

Category	Field	VSI Type						
		PF	VF	VMDq2	VMDq1	Cascaded VEB	Cascaded S-comp	Floating VEB
	Queuing Option Section is Valid	1						
	Scheduler Section is Valid	Zero if non-CDB.						
Switching	Switch ID	If the VSI is connected directly to a port virtualizer, this value must be set. For a port connected to a VEB or directly to the MAC, this value might be left at zero and the VEB switch ID or the default switch ID is used. When connecting to a port virtualizer, the Switch ID should be set to the S-tag of the channel and the <i>Is Not</i> S-tag should be cleared.					Set to the first tag covered by this S-comp	VEB switch ID
	Is Not S-tag						0	1
	Allow Loopback	0	1 if VEB 0 if VEPA		1 if part of a VEB, 0 otherwise ¹	1 if VEB 0 if VEPA	0	1
	Allow Local Loopback	0	0	0	1 if part of a VEB, 0 otherwise	1 if VEB 0 if VEPA	0	0
Security	Allow Destination Override	1	0	1	1	1	1	0/1
	Enable VLAN Anti Spoof	0	1	0	0	0	0 ²	0/1
	Enable MAC Anti Spoof	0	1	0	0	0	0	0/1
VLAN Handling	PVID + Default UP (16 bits)	N/A	Port-based VLAN	N/A	N/A	N/A	N/A	N/A
	VLAN Driver Insertion Mode	11b	Depends on VLAN Awareness	11b	11b	11b	11b	11b
	Insert PVID	0		0	0	0	0	0
	VLAN and UP Expose Mode (Rx)	00b (extract and show)		00b (extract and show)	00b (extract and show)	00b (extract and show)	00b (extract and show)	00b (extract and show)
Ingress UP Translation	Ingress UP Translation Table	Identity mapping (default).						
Egress UP Translation	Egress UP Translation Table	Identity mapping (default).						
Cascaded Port Virtualize	S-tag	Same as Switch ID if port virtualizer is part of the switch; otherwise, not applicable.					N/A	Switch ID of the floating VEB
	S-tag Extract Mode	01b (extract S-tag and do not insert in descriptor).					10b (extract S-tag from packet and expose in descriptor)	01b
	S-tag Insert Enable	1 if port virtualizer is part of the switch topology, 0 otherwise.					0	1
	Prune Based on Internal S-tag Enable	1					0	1
	Accept Tag From Host	0					1	0
Queue Mapping	Mapping Method	Depends on queue mapping method.						
	Queue Mapping							
	Number and Offset of Receive Queues per TCs	According to allocation to TCs.						



Table 7-74. Add VSI Recommended Settings

Category	Field	VSI Type						
		PF	VF	VMDq2	VMDq1	Cascaded VEB	Cascaded S-comp	Floating VEB
	Fragmented Multicast UDP Enable	0 unless used for UDA.					0	0 unless used for UDA.
	Fragmented Unicast UDP Enable	0 unless used for UDA.					0	
	Multicast UDP Packets Enable	0 unless used for UDA.					0	
	Unicast UDP Packets Enable	0 unless used for UDA.					0	
	RSS LUT	PF	VSI	VSI	N/A	N/A	N/A	VSI
Scheduler	Enabled TCs	According to allocated TCs (0x1 for non-DCB environment).						

1. Different VSIs within a VEB might have different loopback modes.
2. VLAN anti spoof does not operate correctly in cascaded S-comp mode.

7.4.9.5.5.2 Update VSI (0x0211)

This command is used to update the parameters of an existing VSI. A function can update only a VSI it controls. Not all the parameters of a VSI can be updated. Only parameters that do not impact the scheduler tree structure can be modified. In order to change the TCs allocated to a VSI, the Configure VSI Bandwidth per Traffic Class command described in [Section 7.8.4.7](#) should be used.

When updating a VSI connected to a port virtualizer in MFP mode, the following restrictions applies:

- The *Switch ID* field should be set to zero.
- The *Cascaded Port Virtualizer section is valid* field should be cleared.
- The S-tag of the VSI can not be modified.

Table 7-75. Update VSI command (Opcode: 0x0211)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0211	Command opcode.
Datalen	4-5	0x80	Length of buffer.
Return Value	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID Number	16-17		VSI SEID number.
Reserved	18-21		Reserved.
Command Flags	22-23		15:0: Reserved.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The command buffer and the completion buffer used to update a VSI is the same as the command and completion buffers of the Add VSI command ([Table 7-71](#) and [Table 7-73](#)). Specific limitations are listed in the table.



All the filters set before the VSI type updates are kept and the software device driver should guarantee they are compliant with the new VSI type.

Table 7-76. Update VSI Response (Opcode: 0x0211)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0211	Command opcode
Datalen	4-5	0x80	Length of buffer
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. The following error values can be returned: ENOENT - if SEID does not point to a valid VSI. EACCES - if the VSI is not owned by this PF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID Number	16-17	0x0	Copied from command.
VSI Number	18-19		Returns the assigned VSI number.
VSI Used	20-21		Number of VSIs used by this function.
Total VSIs <u>Not</u> <u>Allocated</u>	22-23		Total number of VSIs still not allocated and not reserved by any function.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

7.4.9.5.5.3 Get VSI Parameters (0x0212)

This command is used to get the parameters of an existing VSI. A function can query only a VSI it controls.

Table 7-77. Get VSI Parameters Response (Opcode: 0x0212)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0212	Command opcode.
Datalen	4-5	0x80	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. The following error values can be returned: ENOENT - if SEID does not point to a valid VSI. EACCES - if the VSI is not owned by this PF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID Number	16-17		VSI SEID number (copied from command).
VSI Number	18-19		Returns the assigned VSI number.
VSIs Used	20-21		Number of VSIs used by this function.

**Table 7-77. Get VSI Parameters Response (Opcode: 0x0212)**

Name	Bytes.Bits	Value	Remarks
Total VSIs Not Allocated	22-23		Total number of VSIs still not allocated and not reserved by any function.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

Table 7-78. Get VSI Parameters Command (Opcode: 0x0212)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0212	Command opcode.
Datalen	4-5	0x80	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID Number	16-17		VSI SEID number.
Reserved	18-23		Reserved.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

Table 7-79 lists the response buffer received when querying a VSI.

Table 7-79. Get VSI Parameters Response Buffer

Byte/Bit	Description
0-95	Same parameters as listed in Table 7-71 .
96-127	Same parameters as listed in Table 7-73 .

7.4.9.5.6 Port Virtualizer Commands (Opcode 0x022x)

7.4.9.5.6.1 Add Port Virtualizer

This command is used to instantiate a port virtualizer on a port. A port virtualizer can be instantiated only when a single VSI is connected between the MAC and a PF. If additional switching elements are active on this port, they must be first disconnected before the port virtualizer can be added.

When an Add Port Virtualizer command is received, firmware checks if all the requested resources are available and allocates them to the port virtualizer. If part of the resources are not available, the command returns a list of unavailable resources.

**Table 7-80. Add Port Virtualizer Command (Opcode: 0x0220)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0220	Command opcode
Datalen	4-5	0x0	Length of buffer
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Command Flags	16-17	0x0	<p>0 = Port Virtualizer Type. Must be set to zero</p> <p>1 = Forward Unknown S-tag. If set, packets with unknown S-tags are forwarded to the default VSI.</p> <p>2 = Reserved.</p> <p>Note: If bit 1 is set, the port type should be default; otherwise, these bits are ignored.</p> <p>3 = Port type:</p> <ul style="list-style-type: none"> • 0b = Default. • 1b = Control. <p>4:15 = Reserved.</p>
Uplink SEID	18-19	0x0	Defines the SEID to which this port virtualizer should be connected. This can only be the MAC of the port on which this function resides.
Connected VSI SEID	20-21	0x	Defines the SEID of the first VSI connected to this port virtualizer. Port type is defined by the <i>Port Type</i> flag. Should point to a valid VSI connected to the uplink SEID as uplink.
Reserved	22-31		Reserved.

Table 7-81. Add Port Virtualizer Response (Opcode: 0x0220)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0220	Command opcode
Datalen	4-5	0x0	Length of buffer
Return Value/ VFID	6-7		<p>Return value. The following error values can be returned:</p> <p>ENOENT - if the port SEID do not point to a valid element.</p> <p>ENOSPC - if there aren't enough resources to assign a Port Virtualizer.</p> <p>ESRCH - if the operation is not permitted (e.g. MFP mode or virtualization is disabled)</p> <p>EACCES - if the port is not owned by this PF.</p> <p>EINVAL - if a driver tries to create a Port Virtualizer of a type contradicting the previously created Port Virtualizer type.</p>
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.



Table 7-81. Add Port Virtualizer Response (Opcode: 0x0220)

Name	Bytes.Bits	Value	Remarks
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Port Virtualizer SEID/Error Reasons	16-17		If the return value is zero: Bytes 25:24 = Returns the SEID of the port virtualizer. If the return value is ENOSPC, contains a bitmap that indicates which resources are missing: 0 = No port virtualizer left. 1 = Not enough scheduler nodes. 2 = Reserved. 3 = Not enough switching entries. 4-31 = Reserved.
Reserved	18-31		Reserved.

7.4.9.5.6.2 Update Port Virtualizer Parameters

This command is used to modify the parameters of a port virtualizer. In this command, the flags of the port virtualizer can be modified. Note that the default VSI cannot be modified using this command, only the routing of unknown packets.

In order to enable forwarding of unknown packets, a default VSI should be defined beforehand.

Table 7-82. Update Port Virtualizer Command (Opcode: 0x0221)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0221	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Command Flags	16-19	0x0	0 = Reserved (the port virtualizer type cannot be modified on the fly). 1 = Forward unknown S-tag. If set, packets with unknown S-tags are forwarded to the default VSI. 2-31 = Reserved.
Reserved	20-31	0x0	Reserved.

Table 7-83. Update Port Virtualizer Response (Opcode: 0x0221)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0221	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value	6-7		The following error values can be returned: ENOENT - if SEID does not point to a port virtualizer. EACCES - if the port virtualizer is not owned by this PF.



Table 7-83. Update Port Virtualizer Response (Opcode: 0x0221)

Name	Bytes.Bits	Value	Remarks
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31		Reserved.

7.4.9.5.6.3 Get Port Virtualizer Parameters

This command is used to get the parameters of a port virtualizer.

Table 7-84. Get Port Virtualizer Command (Opcode: 0x0222)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0222	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID	16-17		The SEID of the port virtualizer.
Reserved	18-31		Reserved.

Table 7-85. Get Port Virtualizer Response (Opcode: 0x0222)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0222	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		The following error values can be returned: ENOENT - if SEID does not point to a port virtualizer. EACCES - if the port virtualizer is not owned by this PF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-17		The SEID of the requested port virtualizer (reflects the command).
Default Tag	18-19		
Command Flags	20-21	0x0	0 = Port virtualizer type: <ul style="list-style-type: none"> • 0b = S-comp. • 1b = Port extender. 1 = Forward unknown S-tag. If set, packets with unknown S-tags are forwarded to the Default VSI. 2 = Reserved. 2:15: Reserved.
Reserved	22-29	Reserved	Reserved.
Default Port SEID	30-31		Returns the SEID of the default port.



7.4.9.5.7 VEB/VEPA Commands (Opcode 0x023x)

7.4.9.5.7.1 Add VEB

This command is used to instantiate a VEB on a port/port virtualizer. A VEB can be instantiated either as a floating element or can be inserted between an existing VSI and its uplink.

When an Add VEB command is received, firmware checks if all the requested resources are available and allocates them to the VEB. If part of the resources are not available, the command returns a list of unavailable resources.

Note: After a VEB is added, the *Allow Loopback* flag of the original VSI should be set using the Update VSI command.

Table 7-86. Add VEB Command (Opcode: 0x0230)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0230	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Uplink SEID	16-17	0x0	Defines the SEID to which this VEB should be connected. If the <i>Floating VEB</i> flag is set, the uplink element should be null (0x0).
Downlink SEID	18-19	0x0	Defines the SEID of the downlink VSI to which this VEB is connected. Port type is defined by flags. Should point to a valid VSI currently connected to uplink SEID. Should be null (0x0) for floating VEB
Flags.Floating VEB	20.0	0x0	If set, it can only send packets to attached VSIs. If cleared, it can send and receive packets from the LAN.
Flags.Port Type	20.2:20.1	0x0	Defines the type of the VSI defined by the <i>Downlink</i> SEID field. 00b = Reserved. 01b = Default. 10b = Data port. 11b = Reserved. Should be default (01b) for floating VEB.
Flags.Reserved	20.3	0x0	Reserved.
Flags.Disable Statistics Gathering	20.4	0x0	Disable Statistics Gathering. 0x0 = Statistics are gathered for this VEB. 0x1 = Statistics are not gathered for this VEB.
Flags.Reserved	20.5:21	0x0	Reserved.
Enabled TCs	22	0x0	A bitmap of the TCs that should be enabled in this VEB. The TCs enabled should be already enabled in the uplink element (should already have a node in the scheduler in the uplink element).
Reserved	23-31	0x0	Reserved.

**Table 7-87. Add VEB Response (Opcode: 0x0230)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0230	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if the uplink SEID does not point to valid elements. ENOSPC - if there aren't enough resources to assign a VEB. EACCES - if the uplink element is not owned by this PF. ERANGE - a TC not enabled in the uplink element was requested. EPERM - if the command is not permitted (such as virtualization is disabled).
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-21		Reserved
Switch ID	22-23		Assigned switch ID. If connected to an S-channel, this is equal to the S-tag, otherwise, it is the switch ID assigned internally by the firmware.
VEB SEID/Error Reasons	24-25		If the return value is zero: Bytes 25:24 = Returns the SEID of the VEB. If the return value is ENOSPC, contains a bitmap that indicates which resources are missing: 0 = No VEB left. 1 = Not enough scheduler nodes. 2 = Not enough statistics counters. 3 = Not enough switching entries. 4-31 = Reserved.
Statistic Counter	26-27	Statistic Counters Index	Returns an index of the statistics counters block assigned to this VEB. This number is in the 0-15 range and points to the VEB statistics set. Relevant only if the <i>Disable Statistics</i> gathering flag is cleared in the command.
VEBs Used	28-29		Number of VEBs used by this function.
Total VEBs Not Allocated	30-31		Total number of VEBs still not allocated and not reserved by any function.

7.4.9.5.7.2 Get VEB Parameters (0x0232)

This command returns the parameters of the VEB/VEPA.

Table 7-88. Get VEB Parameters Command (Opcode: 0x0232)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0232	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.

**Table 7-88. Get VEB Parameters Command (Opcode: 0x0232)**

Name	Bytes.Bits	Value	Remarks
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID	16-17	0x0	Defines the SEID of the VEB.
Reserved	18-31	0x0	Reserved.

Table 7-89. Get VEB Parameters Response (Opcode: 0x0232)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0232	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if SEID does not point to a valid VEB/VEPA element. EACCES - if the VEB is not owned by this PF. EPERM - if the command is not permitted (such as virtualization is disabled).
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID	16-17		The SEID requested in the command.
Switch ID	18-19		Assigned switch ID. If connected to an S-channel, this is equal to the S-tag, otherwise, it is the switch ID assigned internally by the firmware.
Flags	20-21	0x0	0 = Floating VEB. If set, can only send packets to attached VSIs. If cleared, can send and receive packets from the LAN. 2:1 = Reserved. 3 = L2 filtering is enabled. If set, the L2 filter table should pass in addition to the regular forwarding decision. Relevant only for cloud and UDP cloud NVM images. Reserved otherwise. 15:4 = Reserved.
Statistic Counter	22-23	Statistic Counters Index	Returns an index of the statistics counters block assigned to this VEB.
VEBs Used	24-25		Number of VEBs used by this function.
Total VEBs Not Allocated	26-27		Total number of VEBs still not allocated and not reserved by any function.
Reserved	28-31		Reserved.

7.4.9.5.8 Switch Connectivity Commands (Opcode 0x024x)

7.4.9.5.8.1 Delete Element

This command is used to remove a switching element. A function can only remove an element it controls. The software device driver should make sure every queue in a VSI is disabled before a VSI is removed; however, queue groups tied to the VSI might be removed together with the VSI. It should also make sure the removed element is not tied to any other element before removing it. An exception to this rule is the case of a VEB, that when removed with a single VSI tied to it, is replaced with this VSI.

**Table 7-90. Delete Element Command (Opcode: 0x0243)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0243	Command opcode.
Datalen	4-5	0x0	Reserved.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID	16-17	SEID	The SEID of the element to remove.
Reserved	18-31	0x0	Must be zero.

Table 7-91. Delete Element Response (Opcode: 0x0243)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0243	Command opcode.
Datalen	4-5	0x0	Reserved.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. The following error values can be returned: ENOENT - if SEID does not point to a valid element. EACCES - if the element is not owned by this PF. EBUSY - if the element to remove has more than one uplink element tied to it.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31		Reserved

7.4.9.5.9 Forwarding Table Configuration Commands (Opcode 0x025x)

Note: All the MAC addresses in the forwarding table configuration commands should be given in big Endian format.

7.4.9.5.9.1 Add MAC, VLAN Pair (0x0250)

This command is used to add a set of MAC or MAC, VLAN pairs to a set of VSIs. If one allocation fails due to lack of resources, the Set VSI Promiscuous Modes command ([Section 7.4.9.5.9.5](#)) might be used to enable forwarding of all the packets of a given type to this VSI.

All the VSIs must have the same switch ID.

To only enable receiving untagged packets, a MAC, VLAN = 0 filter should be added.

Note: The ToQueue action might be ignored if the packet is forwarded due to multiple rules match (such as exact match and promiscuous unicast).



Table 7-92. Add MAC, VLAN Pair Command (Opcode: 0x0250)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0250	Command opcode.
Datalen	4-5		Length of buffer. Should be equal to Num_Adresses * 16 bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Num_Adresses	16-17		The number of MAC, VLAN pairs to add.
SEID 0	18-19	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
SEID 1	20-21	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
SEID 2	22-23	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The command buffer of this command contains the details of the MAC, VLAN pairs to add. It contains a set of Num_addresses 16-byte structures as follows.

Field	Offset	Description
MAC Address	0-5	MAC address to add.
VLAN Tag	6-7	VLAN tag to add. To only enable receiving untagged packets, a VLAN ID of zero should be set and the <i>Ignore VLAN</i> flag should be cleared. 23:12 = Reserved. 11:0 = VLAN ID.
Flags	8-9	0 = Use perfect match. If set, a perfect match MAC address might be used to create this MAC, VLAN pair. Must be set. 1 = Reserved. 2 = Ignore VLAN. If set, the VLAN tag is ignored and the MAC address is used to forward packets from all VLANs. 3 = ToQueue. Use MAC, VLAN to point to a queue. This flag should be set only if the filter points to a single VSI. If a filter points to multiple VSIs, an error is returned. 4 = Use shared MAC. If set, the MAC address is taken from the shared pool. 15:0 = Reserved.
Queue Number	10-11	Bit 15:11 = Reserved. Bit 10:0 = Queue number. Valid only if the Flags.ToQueue bit is set. The queue number is relative to the VSI.
Reserved	12-15	Reserved for response part.



Table 7-93. Add MAC, VLAN pair Response (Opcode: 0x0250)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0250	Command opcode.
Datalen	4-5		Length of buffer. Equals to Num_Addresses * 16 bytes.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOSPC - if there aren't enough resources to assign all the MAC, VLAN pairs or an attempt to add a VSI to an existing filter with ToQueue set. The return buffer details which of the allocations failed. ENOENT - if SEID does not point to a valid VSI. EACCES - if the VSI is not owned by this PF. EEXIST - if a queue is assigned to a multicast filter (more than one VSI) or a resource allocated for a dedicated pool is reused. EINVAL - if a ToQueue flag is set for a multicast filter.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Perfect Match MAC Used	16-17		Number of dedicated perfect match MAC used by this function.
Perfect Match MAC Not Allocated	18-19		Total number of perfect match MAC still not allocated and not reserved by any function.
Reserved	20-23		
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The response buffer of this command contains the results of the MAC, VLAN pairs allocation. It contains a set of Num_addresses 16-byte structures as follows.

Field	Offset	Description
Reserved	0-11	Reserved for Command part
Matching Method	12	0x0 = Perfect match was used. 0x1:0xFE = Reserved. 0xFF = Request failed due to lack of resources.
Reserved	13-15	Reserved.

7.4.9.5.9.2 Remove MAC, VLAN Pair (0x0251)

This command is used to remove a set of MAC or a MAC, VLAN pairs from up to 3 VSIs. All the VSIs must have the same switch ID.

Table 7-94. Remove MAC, VLAN Pair Command (Opcode: 0x0251)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0251	Command opcode.
Datalen	4-5		Length of buffer. Should be equal to Num_Addresses * 16 bytes.



Table 7-94. Remove MAC, VLAN Pair Command (Opcode: 0x0251)

Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Num Addresses	16-17		The number of addresses in the command buffer.
SEID 0	18-19	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
SEID 1	20-21	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
SEID 2	22-23	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The command buffer of this command contains the details of the MAC, VLAN pairs to remove. It contains a set of Num_addresses 16-byte structures as follows.

Table 7-95. Remove MAC, VLAN Pair Response (Opcode: 0x0251)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0251	Command opcode.
Datalen	4-5		Length of buffer. Indicate the number of entries filled by firmware (16*Num Addresses) even if Datalen in the command was larger than that.
Return Value/ VFID	6-7		Return value. The following error values can be returned: EINVAL - if all VSIs do not point to the same switch ID. ENOENT- if the MAC, VLAN pair does not exist or if one of the SEID does not point to a valid VSI. The details of which remove request failed is found in the response buffer. Note: If a VSI is not connected to one of the MAC, VLAN to remove, it is silently ignored. EACCES - if one of the VSIs is not owned by this PF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Perfect Match MAC Used	16-17		Number of dedicated perfect match MAC used by this function.
Perfect Match MAC Not Allocated	18-19		Total number of perfect match MAC still not allocated and not reserved by any function.
Reserved	20-23		Reserved.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		



Field	Offset	Description
MAC Address	0-5	MAC address to remove the VSI from.
VLAN tag	6-7	VLAN tag to remove the VSI from.
Flags	8	0 = Perfect match. If set, a perfect match MAC address was used to create this MAC, VLAN pair. 2:1 = Reserved. 3 = Ignore VLAN. If set, the forwarding is currently based only on MAC. 4 = Remove from all VSIs. This removes the filter from all the VSIs owned by this PF. 7:5 = Reserved.
Reserved	9-15	Reserved.

The response buffer of this command contains the results of the MAC, VLAN pairs removal. It contains a set of Num_addresses 16-byte structures as follows.

Field	Offset	Description
Reserved	0-11	Reserved for command part.
Error code	12	0x0 = Successful removal. 0x1:0xFE = Reserved. 0xFF = Request failed (MAC, VLAN pair does not exist).
Reserved	13-15	Reserved.

7.4.9.5.9.3 Add VLAN (0x0252)

This command is used to add a set of VLANs to the VLAN table or to add a set of VLAN filters to up to 3 VSIs. All the VSIs must have the same switch ID.

Note: In order to support transmit VLAN filtering, the *Enable VLAN anti spoof* flag should be set in the relevant Add VSI command. In order to support receive VLAN filtering, the *Promiscuous VLAN* flag in the Set VSI Promiscuous Modes command should be cleared.

For PVLAN functionality, both ingress and egress VLAN filtering are needed.

Note: By default, VLAN filtering is disabled. The first time this command is applied, VLAN filtering is enabled.

Table 7-96. Add VLAN Command (Opcode: 0x0252)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0252	Command opcode.
Datalen	4-5		Length of buffer. Should be equal to Num_VLAN * 8 bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Num_VLAN	16-17		Number of VLANs to add.



Table 7-96. Add VLAN Command (Opcode: 0x0252)

Name	Bytes.Bits	Value	Remarks
SEID 0	18-19	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
SEID 1	20-21	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
SEID 2	22-23	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The command buffer of this command contains the details of the VLAN IDs to add. It contains a set of Num_VLAN 8-byte structures as follows.

Table 7-97. Add VLAN Response (Opcode: 0x0252)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0252	Command opcode.
Datalen	4-5		Length of buffer. Equals to Num_VLAN * 8 bytes.
Return Value/ VFID	6-7		Return value. The following error values can be returned: EINVAL - if all the VSIs do not point to the same switch ID. ENOSPC - if there aren't enough resources to assign the VLANs. The response buffer details which of the allocations failed. ENOENT - if one of the VSIs is not valid. EACCES - if one of the VSIs is not owned by this PF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-19		Reserved.
VLAN Used	20-21		Total number of VLANs used (VLAN is a shared resource).
VLAN Not Allocated	22-23		Total number of VLAN still not allocated.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		



Field	Offset	Description
VLAN Tag	0-1	VLAN tag to add.
Flags	2	0 = Local VLAN. 2:1 = PVLAN type: <ul style="list-style-type: none"> • 00b = Regular VLAN (not private). • 01b = Primary VLAN. • 10b = Secondary VLAN. • 11b = Reserved. 4:3 = PVLAN port type (valid only if PVLAN type is primary or secondary VLAN): <ul style="list-style-type: none"> • 00b = Regular VLAN. • 01b = Promiscuous VSIs. • 10b = Community VSIs. • 11b = Isolated VSIs. 7:5 Reserved. Note: If primary or secondary VLAN is set, the port type cannot be zero. If both are not set, it must be zero. The definition of a VLAN ID should be consistent within a VEB. Thus, subsequent commands should set the same PVLAN type and local VLAN flags when adding VSIs to the same VLAN ID. The PVLAN port type can be different for various VSIs within the same VLAN.
Reserved	3-7	Reserved for response part.

The response buffer of this command contains the results of the VLANs allocation. It contains a set of Num_VLAN 8-byte structures as follows.

Field	Offset	Description
Reserved	0-3	Reserved for command part.
Result	4	0x0 = Allocation success. 0x1:0xFD = Reserved. 0xFE = Request failed due to inconsistent VLAN definition. 0xFF = Request failed due to lack of resources.
Reserved	5-7	Reserved.

7.4.9.5.9.4 Remove VLAN (0x0253)

This command is used to remove a set of VLANs from the VLAN table or to remove VLAN filters from up to 3 VSIs.

All the VSIs must have the same switch ID.

Table 7-98. Remove VLAN Command (Opcode: 0x0253)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0253	Command opcode.
Datalen	4-5	0x0	Length of buffer. Should be equal to Num_VLAN * 8 bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.



Table 7-98. Remove VLAN Command (Opcode: 0x0253)

Name	Bytes.Bits	Value	Remarks
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Num_VLAN	16-17		Number of VLANs to add.
SEID 0	18-19	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
SEID 1	20-21	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
SEID 2	22-23	0x0	Bit 15 = Valid SEID. Bit 14:10 = Reserved. Bit 9:0 = SEID Number of the VSI.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The command buffer of this command contains the details of the VLAN pairs to remove. It contains a set of Num_VLAN 8-byte structures as follows.

Field	Offset	Description
VLAN Tag	0-1	VLAN tag to remove VSI(s) from.
Flags	2	0 = Remove entire VLAN. 7:1 = Reserved.
Reserved	3	Reserved.
Reserved	4-7	Reserved for response part.

Table 7-99. Remove VLAN Response (Opcode: 0x0253)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0253	Command opcode.
Datalen	4-5		Length of buffer. Equals to Num_VLAN * 8 bytes.
Return Value/ VFID	6-7		Return value. The following error values can be returned: EINVAL - if all the VSIs do not point to the same switch ID. ENOENT - if the VLAN does not exist or if one of the SEID does not point to a valid VSI. The response buffer details which of the removal failed. Note: If a VSI is not connected to one of the MAC, VLAN to remove, it is silently ignored. EACCES - if one of the VSIs is not owned by this PF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-19		Reserved.
VLAN Used	20-21		Total Number of VLANs used (VLAN is a shared resource).

**Table 7-99. Remove VLAN Response (Opcode: 0x0253)**

Name	Bytes.Bits	Value	Remarks
VLAN Not Allocated	22-23		Total number of VLANs still not allocated.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The response buffer of this command contains the results of the MAC, VLAN pairs removal. It contains a set of Num_VLAN 8-byte structures as follows.

Field	Offset	Description
Reserved	0-3	Reserved for command part.
Error Code	4	0x0 = Successful removal. 0x1:0xFE = Reserved. 0xFF = Request failed (the VLAN does not exist or if one of the VSIs is not valid).
Reserved	5-7	Reserved

7.4.9.5.9.5 Set VSI Promiscuous Modes (0x0254)

This command is used to enable a VSI to set various promiscuous modes and other generic filtering options.

Note: If the *Default VSI* flag is set for a VSI within a VEB, this command changes the default VSI of the VEB to the VSI pointed by this command.

Table 7-100. Set VSI Promiscuous Modes Command (Opcode: 0x0254)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0254	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.



Table 7-100. Set VSI Promiscuous Modes Command (Opcode: 0x0254)

Name	Bytes.Bits	Value	Remarks
Flags	16-17		<p>0 = Promiscuous unicast. 1 = Promiscuous multicast. 2 = Promiscuous broadcast. 3 = Default VSI. Accept packets within the switch ID not matching any specific address to this VSI. 4 = Promiscuous VLAN. 14:5 = Reserved. 15 = Apply promiscuous mode to Rx traffic only. Note: This bit is relevant only if one of options 0, 1, 2 is used.</p> <p>This bit is relevant only if the switch is configured to operate in limited promiscuous mode and packet is not hitting a VLAN or ingress mirroring rule.</p> <p>Multiple flags might be set. Note: Default VSI is supposed to be used only for VMDq1 scenarios where there is no VEB. If this command is used in the presence of a VEB, it changes the default VSI of the VEB.</p> <p>Default VSI should not be set if the VSI is connected directly to the port (not via a VEB or a PV).</p>
Valid Flags	18-19		<p>0 = Promiscuous unicast flag is valid. 1 = Promiscuous multicast is valid. 2 = Promiscuous broadcast is valid. 3 = Default VSI is valid. 4 = Promiscuous VLAN is valid. 14:6 = Reserved. 15 = Apply promiscuous mode to Rx traffic only if the flag is valid. Note: If not set, assume promiscuous traffic is applied to Tx and Rx (legacy behavior) Multiple valid bits might be set.</p>
SEID Number	20-21		<p>Bit 15 = Valid SEID. Ignored in this field. Bit 14:10 = Reserved. Bit 9:0 = SEID number of the VSI.</p>
VLAN ID	22-23		<p>15 = VLAN is valid. 14:12 = Reserved. 11:0 = VLAN ID. Note: If bit 15 is set, the <i>Promiscuous Unicast</i>, <i>Multicast</i>, and <i>Broadcast</i> flags apply only to this VLAN; otherwise, these modes apply to all VLANs.</p> <p>If VSI is in promiscuous VLAN mode, the VLAN ID should not be used.</p>
Reserved	24-31	0x0	Reserved.

Table 7-101. Set VSI Promiscuous Modes Response (Opcode: 0x0254)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0254	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - If the SEID doesn't point to a valid VSI. ENOSPC - if there aren't enough resources to apply the promiscuous rules. EACCESS - if the VSI is not owned by this PF.

**Table 7-101. Set VSI Promiscuous Modes Response (Opcode: 0x0254)**

Name	Bytes.Bits	Value	Remarks
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31		Reserved. Should contain the parameters from the command.

7.4.9.5.9.6 Add S-tag (0x0255)

This command is used to associate an S-tag with a VSI. This command can be given only by the port controlling the port virtualizer. This command should be used for cascaded port virtualizer VSIs where more than one S-tag points to the same VSI. For regular channels, the Add VSI command already adds the single S-tag of the VSI.

An S-tag might be directed to a specific queue in the VSI. This is done by setting the *ToQueue* flag and providing a valid queue in the *QueueNumber* field. In order to assign a queue to the default VSI, this command should be used with the default S-tag as tag value.

Note: The first S-tag added to the cascaded port virtualizer VSI via the Add VSI command is considered as the switch ID for this VSI and should not be manipulated by the Add S-tag and Remove S-tag commands.

Note: VEB or L2 filters cannot be applied to a cascaded VSI on any S-tag. The VSI should be in default mode for its initial S-tag. The VSI should be defined with a Connection Type of Default Port.

Table 7-102. Add S-tag Command (Opcode: 0x0255)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0255	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7	0x0	Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Flags	16-17		0 = ToQueue. 15:1 = Reserved.
SEID	18-19	0x0	Bit 15 = Valid SEID. Ignored in this field. Bit 14:10 = Reserved. Bit 9:0 = SEID number of the VSI. Defines the VSI that uses this S-tag. This VSI should belong to the port virtualizer through which this command is received.
Tag	20-21	0x0	The value of the S-tag.
Queue Number	22-23	0x0	Bit 15:11 = Reserved. Bit 10:0 = Queue number. Valid only if the Flags.ToQueue bit is set. The queue number is relative to the VSI.
Reserved	24-31		Reserved

**Table 7-103. Add S-tag Response (Opcode: 0x0255)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0255	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if SEID does not point to a valid element. ENOSPC - if there aren't enough resources to assign a tag. EACCES - if the VSI is not owned by this PF. EEXIST - if this tag already points to another VSI. EPERM - attempt to add tags to a VSI which is not a cascaded type.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-27	0x0	Reserved.
S-tags used	28-29		Number of tags used by this PF.
S-tags Not Allocated	30-31		Total number of tags still not allocated.

7.4.9.5.9.7 Remove S-tag (0x0256)

This command is used to remove an S-tag from the forwarding table of a VSI.

Note: This command should be used only to remove tags in a cascaded port virtualizer VSIs.

Table 7-104. Remove S-tag Command (Opcode: 0x0256)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0256	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
VSI SEID	16-17	0x0	Bit 15 = Valid SEID. Ignored in this field. Bit 14:10 = Reserved. Bit 9:0 = SEID number of the VSI. Defines the VSI SEID that uses this tag. This VSI should belong to the port virtualizer through which this command is received.
S-tag	18-19	0x0	The value of the tag.
Reserved	20-31	0x0	Reserved.

**Table 7-105. Remove S-tag Response (Opcode: 0x0256)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0256	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if SEID does not point to a valid element. EACCES - if the VSI is not owned by this PF. ENXIO - this S-tag does not point to this VSI. EPERM - attempt to remove tags from a VSI which is not a cascaded type.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-27		Reserved.
S-tag used	28-29		Number of S-tag used by this function.
S-tag Not Allocated	30-31		Total number of S-tag still not allocated.

7.4.9.5.9.8 Update S-tag (0x0259)

This command is used to update the S-tags associated with a VSI. This command can be given only by the port controlling the port virtualizer. This command can be used only for cascaded port virtualizer VSIs.

If a queue was associated with the previous tag, the association is kept for the new tag.

Table 7-106. Update S-tag Command (Opcode: 0x0259)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0259	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7	0x0	Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID Number	16-17	0x0	Bit 15 = SEID Valid. Ignored in this field. Bit 14:10 = Reserved. Bit 9:0 = Defines the SEID of the VSI that uses this tag. This VSI should belong to the PF through which this command is received.
Old Tag	18-19	0x0	The original value of the S-tag. If the value is zero, it is assumed the VSI didn't have a tag.
New Tag	20-21	0x0	The new value of the tag.
Reserved	22-31		Reserved.

**Table 7-107. Update S-tag Response (Opcode: 0x0259)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0259	Command opcode
Datalen	4-5	0x0	Length of buffer
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if SEID does not point to a valid element. ENOSPC - if there aren't enough resources to assign a new tag. EACCES - if the VSI is not owned by this PF. EEXIST - if the new tag already points to another VSI. EPERM - Attempt to modify a tags in a VSI which is not a cascaded port extender type.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-27	0x0	Reserved.
S-tags Used	28-29		Number of S-tags used by this PF.
S-tags Not Allocated	30-31		Total number of S-tags still not allocated.

7.4.9.5.9.9 Add Control Packet Filter (0x025A)

This command is used to add a control filter to forward packets to a control VSI. This function should be used to forward packets to control VSIs; however, there is no enforcement of this rule and the software device driver might use it to forward control packets to other VSIs. Only packets reaching the switching element is forwarded by this rule. This means that:

- If the VSI is connected as a control port of the MAC, a port virtualizer or a VEB directly connected to the MAC, this filter applies to untagged packets (no S-tag).
- If the VSI is connected to an S-channel, either directly or as a VEB port, only a packet tagged with the S-channel tag is forwarded.

Note: These filters are exclusive. If a request to set a filter on an existing flow type is received, it is rejected with an EEXIST reason code. If a filter is set to forward an Ethertype ignoring the MAC address (Ignore MAC = 1b), a filter with the same Ethertype and a MAC address should not be applied within the same switch.

Note: The Ethertype programmed by this command should not be one of the L2 tags Ethertype (VLAN, S-tag, etc.) and should not be IP or IPv6.

Table 7-108. Add Control Packet Filter Command (Opcode: 0x025A)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See for details.
Opcode	2-3	0x025A	Command opcode.
Datalen	4-5		Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.



Table 7-108. Add Control Packet Filter Command (Opcode: 0x025A)

Name	Bytes.Bits	Value	Remarks
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
MAC Address	16-21		The MAC address to use in the filter.
Ethertype	22-23		The Ethertype to use. The 16-bit value of the Ethertype.
Command Flags	24-25		0 = Ignore MAC. If set, forwarding is based only on Ethertype. 1 = Drop filter. If set, packets received or sent with this destination MAC address and Ethertype are dropped. 2 = ToQueue. If set, the packets matching this filter is sent to a specific queue. Note: If cleared, the regular queuing mechanism is used. A queue defined by a lower priority switch filter (such as MAC filter) is ignored. 3 = Direction: <ul style="list-style-type: none"> • 0b = Apply to Rx traffic. • 1b = Apply to Tx traffic. 15:4 = Reserved.
SEID Number	26-27		Bit 15 = SEID valid. Ignored in this field. Bit 14:10 = Reserved. Bit 9:0 = Defines the SEID of the VSI that should get the packet. Note: This field is ignored if the <i>Drop Filter</i> flag is set.
Queue Number	28-29		Queue to send the packet to if the <i>ToQueue</i> flag is set. The queue number is relative to the VSI.
Reserved	30-31		Reserved.

Table 7-109. Add Control Packet Filter Response (Opcode: 0x025A)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x025A	Command opcode.
Datalen	4-5		Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOSPC - if there aren't enough resources to assign the requested filter. ENOENT - if the VSI is not valid. EACCES - if the VSI is not owned by this PF. EEXIST - if such a filter already exists and is allocated to another VSI.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
MAC, Ethertype Used	16-17		Number of perfect match MAC, Ethertype used by the device.
Ethertype Used	18-19		Number of perfect Ethertype used by the device.
MAC, Ethertype Not Allocated	20-21		Number of perfect match MAC, Ethertype still not allocated.
Ethertype Not Allocated	22-23		Number of perfect Ethertype still not allocated.
Reserved	24-31		Reserved.



7.4.9.5.9.10 Remove Control Packet Filter (0x025B)

This command is used to remove a control filter. The filter to remove is defined by the switching element to which the VSI is connected. This means that:

- If the VSI is connected as a control port of the MAC, a port virtualizer or a VEB directly connected to the MAC, the removed filter relates to untagged packets (no S-tag).
- If the VSI is connected to an S-channel, either directly or as a VEB port, the removed filter relates to packet tagged with the S-channel tag.

Table 7-110. Remove Control Packet Filter Command (Opcode: 0x025B)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x025B	Command opcode.
Datalen	4-5		Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
MAC Address	16-21		The MAC address in the filter.
Ethertype	22-23		The Ethertype used. The 16-bit value of the Ethertype.
Command Flags	24-25		0 = Ignore MAC. If set, forwarding is based only on Ethertype. 2:1 = Reserved. 3 = Direction: • 0b = Apply to Rx traffic. • 1b = Apply to Tx traffic. 15:4 Reserved.
SEID Number	26-27		Bit 15 = SEID valid Ignored in this field. Bit 14:10 = Reserved. Bit 9:0 = Defines the SEID of the VSI to which the filter is associated.
Reserved	28-31		Reserved.

Table 7-111. Remove Control Packet Filter Response (Opcode: 0x025B)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x025B	Command opcode.
Datalen	4-5		Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT- if the VSI is not connected to the MAC, Ethertype or Ethertype filter to remove. EACCES - if the VSI is not owned by this PF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
MAC, Ethertype Used	16-17		Number of perfect match MAC, Ethertype used by the device.



Table 7-111. Remove Control Packet Filter Response (Opcode: 0x025B)

Name	Bytes.Bits	Value	Remarks
Ethertype Used	18-19		Number of perfect Ethertype used by the device.
MAC, Ethertype Not Allocated	20-21		Number of perfect match MAC, Ethertype still not allocated.
Ethertype Not Allocated	22-23		Number of perfect Ethertype still not allocated.
Reserved	24-31		Reserved.

7.4.9.5.9.11 Add Cloud Filters (0x025C)

This command is used to add a set of cloud filters to a VSI. The filters set by this command are the filters specific to cloud formats. To add MAC or MAC, VLAN filters, the Add MAC, VLAN pairs (0x0250) command should be used.

Note: As the response cannot contain all the cloud resources available, the software device driver needs to use the Get Switch Resources Allocation command (0x0204) to get the resources left after allocation of a filter (Section 7.4.9.5.3.5).

Note: (A0) These filters are available only for VSIs defined as Cloud VSIs in the Add VSI or Update VSI commands (Flags.CloudVSI = 1).

Table 7-112. Add Cloud Filters Command (Opcode: 0x025C)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x025C	Command opcode
Datalen	4-5		When big buffer = 0, equals to 64 * Num_filters bytes. When big buffer = 1, equals to 128 * Num_filters bytes.
Return Value/ VFID	6-7		Return value. Zeroed by driver. Written by Firmware
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Num_filters	16		The number of filters to add
Reserved	17	0x0	Reserved
SEID	18-19	0x0	Bit 15 = SEID valid. Ignored in this field (SEID is always valid). Bit 14:10 = Reserved. Bit 9:0 = Defines the SEID of the VSI.
Big Buffer	20.0		When set, the command buffer is 128 bytes and contains an extra 64 bytes for custom filtering.
Reserved	20.1-23	0x0	Reserved.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The command buffer of this command contains the details of the filters to add. It contains a set of Num_filters structures as follows.



Table 7-113. Add Cloud Filters Response (Opcode: 0x025C)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x025C	Command opcode.
Datalen	4-5		Length of buffer. Equals to 64 * Num_filters bytes.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOSPC - if there aren't enough resources to assign all the filters. The return buffer details which of the allocations failed. ENOENT - if the VSI is not valid. EACCES - if the VSI is not owned by the offload PF. EEXIST - This filter is already allocated to another VSI or a resource allocated for a dedicated pool is reused.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Num_filters	16		The number of filters in the response buffer.
Reserved	17-23		Reserved.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

Table 7-114. Add Cloud Filters - Command Buffer

Field	Offset	Description
Outer MAC Address	0-5	Outer MAC Address to add.
Inner MAC Address	6-11	Inner MAC address to add.
Inner VLAN Tag	12-13	Inner VLAN tag to add (only 12 LSBs are relevant to filter).
IP	14-29	IP address to add. For IPv4 addresses, bytes 14-25 are reserved.



Table 7-114. Add Cloud Filters - Command Buffer

Field	Offset	Description
Command Flags	30-31	<p>6:0 = Filter type:</p> <ul style="list-style-type: none"> • 0x0 = Reserved. • 0x1 = Reserved. • 0x2 = Reserved. • 0x3 = Inner MAC, inner VLAN (for NVGRE, VXLAN, Geneve or VXLAN-GPE packets). • 0x4 = Inner MAC, inner VLAN, tenant ID (for NVGRE, VXLAN, Geneve or VXLAN-GPE packets). • 0x5 = Reserved or VXLAN-GPE • 0x6 = {Inner MAC, tenant ID} (NVGRE packet, VXLAN, Geneve or VXLAN-GPE packets). • 0x7 = Reserved • 0x8 = Reserved • 0x9 = Outer MAC L2 filter. • 0xA = Inner MAC filter. • 0xB = Outer MAC, tenant ID and inner MAC. • 0xC = Application destination IP. • 0xD:0xF = Reserved. • 0x10 = L4 port filter (Mode 1) / {Destination IP, L4 port} filter (Modes 2 and 3). • 0x11 = {Outer MAC, L4 port} filter (Mode 2) / {Inner MAC, L4 port} filter (Mode 3). • 0x12 = {Outer MAC, outer VLAN, L4 port} filter (Mode 2) / {Inner MAC, inner VLAN, L4 port} filter (Mode 3). • 0x13:0x7F = Reserved. <p>Note: Filter 0xB should not be used in modes with port extenders.</p> <p>An outer MAC L2 filter (0x9) should be added only once per relevant MAC address. This filter applies to all the VSI of type cloud.</p> <p>7 = ToQueue. If set, a packet matching this filter is sent to a specific queue. Not relevant if filter type = 0x9.</p> <p>8 = IP address type:</p> <ul style="list-style-type: none"> • 0b = IPv4. • 1b = IPv6. <p>Relevant only for filter type 0xC.</p> <p>12:9 = Tunnel type:</p> <ul style="list-style-type: none"> • 0x0 = VXLAN. • 0x1 = NVGRE or other MAC in GRE. • 0x2 = Geneve. • 0x3 = IP in GRE. • 0x4 = Reserved. • 0x6-0xF = Reserved. <p>Relevant only for filter types 0x4, 0x6, and 0xB.</p> <p>13 = Use shared outer MAC. If set, the outer MAC is taken from the shared pool. Relevant only for filters involving the outer MAC.</p> <p>14 = Use shared inner MAC.: If set, the inner MAC address is taken from the shared pool. Relevant only for filters involving the inner MAC.</p> <p>15 = Use shared Outer IP: If set, the Outer IP address is taken from the shared pool. Relevant only for Filters involving Outer IP.</p>
Tenant ID	32-35	<p>Key to add:</p> <ul style="list-style-type: none"> • NVGRE tunnel type = TNI. • VXLAN/Geneve/VXLAN-GPE tunnel type = VNI. • Other GRE tunnel type = GRE key. <p>Note: The 3-byte tenant ID should be placed in bytes 34:32, except for Geneve on which the VNI should be placed in bytes 35:33.</p>
Reserved	36-39	Reserved.
Queue Number	40-41	<p>Bit 15:11 = Reserved.</p> <p>Bit 10:0 = Queue number. Valid only if the Command Flags.ToQueue bit is set. The queue number is relative to the VSI.</p>
Reserved	42-55	Reserved.
Reserved	56-63	Reserved for response part.
General Fields	64-127	<p>Fields exist only when the big buffer is set.</p> <p>Used for inputs for non-default filters, which were created using the Set Switch Configuration command.</p>



The response buffer of this command contains the results of the filters allocation. It contains a set of Num_filters 64-byte structures as follows.

Field	Offset	Description
Reserved	0-55	Reserved for command part.
Allocation Result	56	0x0 = Filter assigned. 0x1:0xF = Reserved. 0xFF = Request failed due to lack of resources.
Reserved	57-63	Reserved.

7.4.9.5.9.12 Remove Cloud Filters (0x025D)

This command is used to remove a set of cloud filters from a VSI. The filters removed by this command are the filters specific to cloud formats. To remove MAC or MAC, VLAN filters, the Remove MAC, VLAN pairs (0x0251) command should be used.

Table 7-115. Remove Cloud Filters Command (Opcode: 0x025D)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x025D	Command opcode.
Datalen	4-5		When big buffer = 0, equals to 64 * Num_filters bytes. When big buffer = 1, equals to 128 * Num_filters bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Num_filters	16		The number of filters to remove
Reserved	17	0x0	Reserved
SEID	18-19	0x0	Bit 15 = SEID Valid. Ignored in this field. Bit 14:10 = Reserved. Bit 9:0 = Defines the SEID of the VSI.
Big Buffer	20.0		When set, the command buffer is 128 bytes and contains an extra 64 bytes for custom filtering.
Reserved	20.1-23	0x0	Reserved.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The command buffer of this command contains the details of the filters to remove. It contains a set of Num_filters 16-byte structures as defined in the Add Cloud filters command buffer ([Table 7-114](#)).

Note: The *ToQueue* flag and *Queue Number* field are not used by the Remove Cloud Filter command.



Table 7-116. Remove Cloud Filters Response (Opcode: 0x025D)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.2 for details.
Opcode	2-3	0x025D	Command opcode
Datalen	4-5		Length of buffer - equals to 64 * Num_filters bytes.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT- if the VSI is not connected to one of the cloud filters or the cloud filter entry does not exist or if one of the VSIs is not valid. The details of which remove request failed is found in the response buffer. EACCES - if the VSI is not owned by the offload PF (VSI is on another port).
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Num_filters	16		The number of filters in the response buffer.
Reserved	17-23	0x0	Reserved.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The response buffer of this command contains the results of the filters allocation. It contains a set of Num_filters 16-byte structures as follows.

Field	Offset	Description
Reserved	0-13	Reserved for command part.
Allocation Result	14	0x0 = Successful removal. 0x1:0xFE = Reserved. 0xFF = Request failed (one of the VSIs is not connected to this filter or the filter entry does not exist).
Reserved	15	Reserved.

7.4.9.5.9.13 Replace Cloud Filters (0x025F)

This command is used to re-configure the internal switch by replacing default cloud filters with custom filters.

Note: New filters should be programmed after any core reset.

New filters are applicable to all functions.

Replace filters can be done only at initialization.

Table 7-117. Replace Cloud Filter Command (Opcode: 0x025F)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x025F	Command opcode.
Datalen	4-5		32.



Table 7-117. Replace Cloud Filter Command (Opcode: 0x025F)

Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. 0 = No error. 1 = Error.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Valid Flags	16		16.0 = Replace cloud: 0b = Replace l1 filter. 1b = Replace cloud filter. 16.1 = Reserved. 16.2 = Mirror cloud filter: 0b = Normal cloud filter. 1b = Mirror cloud filter. Packets hitting this filter goes to the destination VSI in addition to the original destination VSI. 16.3 = High priority cloud filter: 0b = Normal priority cloud filter. Packets hitting this filter and other cloud filters go to both destination VSIs. 1b = High priority cloud filter. Packets hitting this filter and other cloud filters only go to high-priority filter destination VSI. 16.4:16.7 = Reserved.
Old Filter	17		Old filter to replace. For L1 filters, the possible values of default filters are: 10 = Stag_Inner_Vlan - //stag + inner frame VLAN. 11 = Tunnel_Key. 12 = Inner_MAC. 14 = IP_DA (big FLU). 15 = Outer_IP_DA (big FLU). For cloud filters, the filter types are described in the Add Cloud Filter command except L2 outer MAC (0x09).
New Filter	18		New filter type used. When replacing cloud filters, the range is 0x10-0x17. When replacing L1 filters, the range is 0x10-0x14 for regular FLUs, 0x16-0x17 for big FLUs. If the new filter type equals the old filter type, entries are updated for that filter. This can be done only if this filter type is configured.
TR Index 1	19		Relevant TR bit index that should be set in order to hit this filter. Relevant only in l1 replace. Bits 0:5 = TR index. Bit 6 = Reserved. Bit 7 = TR Index valid. If both TR index 1 and 2 are valid, an OR operation is used for the filter.
TR Index 2	20		Relevant TR bit index that should be set in order to hit this filter. Relevant only in l1 replace. Bits 0:5 = TR index. Bit 6 = Reserved. Bit 7 = TR Index valid. If both TR index 1 and 2 are valid, an OR operation is used for the filter.
Reserved	21-23	0x0	Reserved.
Data Address High	24-27		Buffer address.
Data Address Low	28-31		

The command buffer of this command contains the details of the filter to configure.



Table 7-118. Replace Cloud Filter Command Buffer (Opcode: 0x025F)

Field	Offset	Description
Entries	0-31	When bug FLUs are defined N=8, otherwise N=3. For entry 'n' (n=0..8): 4*N.0 - 4*N.5 : for I1 - Field vector index, for cloud - I1 index: 4*N.6 0 = FV index. Input codes are as follows: 0 = MAC_DA= 0. 6 = Stag_Ethertype. 7 = Stag. 8 = VLAN //single or inner VLAN of outer frame only. 10 = Stag_Inner_Vlan = 10 //stag+ inner frame VLAN. 11 = Tunnel_Key. 12 = Inner_MAC. 14 = IP_DA (big FLU). 15 = Outer_IP_DA (big FLU). 16 = FLU number 0x10 (new, if added in I1 replace). ... 23 = FLU number 0x17 (big FLU. New, if added in I1 replace). 4*N.7: Valid bit. 4*N+1: Reserved. 4*N+2-4*N+3: 32 bit mask. Not relevant for big FLU and cloud.

Table 7-119 Replace Cloud Filters Response Buffer (Opcode: 0x025F)

Field	Offset	Description
Entries	0-31	For entry 'n' (N=0..8): 4*N = Filter index. 4*N+1 = First input, MSB is valid. 4*N+2 = Second input, MSB is valid. 4*N+3 = Third input, MSB is valid.

7.4.9.5.10 Mirroring Commands (Opcode 0x026x)

The mirroring behavior is described in [Section 7.4.6.2.1](#).

7.4.9.5.10.1 Add Mirror Rule (0x0260)

This command is used to add a mirror rule to a specific switch. Mirror rules are supported for VEBs or VEPA elements only.

Table 7-120. Add Mirror Rule Command (Opcode: 0x0260)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0260	Command opcode
Datalen	4-5		Length of buffer - should be equal to 2 * Number of mirrored entries.



Table 7-120. Add Mirror Rule Command (Opcode: 0x0260)

Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID	16-17		Defines the SEID of the switch to which the rule refers.
Rule Type	18-19		2:0 = Rule type: <ul style="list-style-type: none"> • 000b = Reserved. • 001b = Virtual port ingress mirroring. • 010b = Virtual port egress mirroring. • 011b = VLAN mirroring. • 100b = All ingress traffic (to this switch). Includes traffic sent from all VSIs connected to this switch. Does not include traffic received from LAN. • 101b = All egress traffic (from this switch). Includes traffic sent to all VSIs in this switch. Does not include traffic sent to the LAN. • 11xb = Reserved. 15:3: Reserved Note: If the rule type is 100b (all ingress) or 101b (all egress), there is no associated buffer. The flags in bytes 0-1 should be set accordingly. VSIs added to a switch after the all ingress/all egress rule was added are not included in the rule and the rule should be removed and re-applied to include them.
Number of Mirrored Entries	20-21		Defines the number of VSI/VLANs that should be mirrored. The values are in the command buffer.
Destination VSI	22-23		Defines the VSI SEID to which the packets matching the mirror rule is mirrored.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

The command buffer is built as number of mirrored entries of two bytes each containing a single VSI/VLAN, depending on the rule requested as listed in [Table 7-121](#).

Table 7-121. Add Mirror Rule Command Buffer

Byte	Description
0-1	Mirrored VSI (SEID) or VLAN ID. For all rule types accept ingress VLAN mirroring, the values are SEIDs of VSI. For Ingress VLAN mirroring rule type, the values are VLAN IDs. The VSIs in the list should be part of the switch defined by the SEID.

[Table 7-122](#) lists the add mirror rule response (with no buffer).

Table 7-122. Add Mirror Rule Response (Opcode: 0x0260)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0260	Command opcode.
Datalen	4-5		Length of buffer. Equals to 2 * Number of mirrored entries.



Table 7-122. Add Mirror Rule Response (Opcode: 0x0260)

Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if the mirror SEID does not point to a valid switch element or the SEID of the mirrored VSI does not point to a valid VSI. ENOSPC - if there aren't enough resources to assign an mirror rule. EACCES - if the VEB is not owned by this PF. EINVAL - if the same VLAN mirroring rule is added twice. EEXIST - rule already assigned. The rule should be removed before being assigned again.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-17		Reserved.
Rule ID	18-19		Defines the rule ID that is returned in the receive descriptor. This number is assigned by firmware and should be used as a handle when requesting a delete of an existing rule. The rule ID is not relevant for the ingress VLAN mirroring rule type (011b).
Mirror Rules Used	20-21		Number of mirror dedicated rules used by this function.
Mirror rules Not Allocated	22-23		Total number of mirror rules still not allocated.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		

7.4.9.5.10.2 Delete Mirror Rule (0x0261)

This command is used to delete an existing mirror rule. If the rule to remove is an ingress VLAN mirroring type, a buffer should be provided containing the currently mirrored VLAN to remove.

Table 7-123. Delete Mirror Rule Command (Opcode: 0x0261)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0261	Command opcode.
Datalen	4-5	0x0	Length of buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
SEID	16-17		Defines the SEID of the switch to which the rule refers.
Rule Type	18-19		2:0 = Rule type: <ul style="list-style-type: none"> • 000b = Reserved. • 001b = Virtual port ingress mirroring. • 010b = Virtual port egress mirroring. • 011b = Ingress VLAN mirroring. • 100b = All ingress traffic (to this switch). • 101b = All egress traffic (from this switch). • 11xb = Reserved. 15:3 = Reserved.
Number of Mirrored Entries	20-21		Defines the number of VSI/VLANs that should be mirrored. The values are in the command buffer.



Table 7-123. Delete Mirror Rule Command (Opcode: 0x0261)

Name	Bytes.Bits	Value	Remarks
Rule ID	22-23		Defines the rule ID that is returned in the receive descriptor. This ID identifies the rule to delete. This ID is the number returned from the add mirror rule response. Relevant only if the rule type is not Ingress VLAN mirroring (011b).
Data Address High	24-27		Address of buffer. Relevant only if the rule type is Ingress VLAN mirroring (011b).
Data Address Low	28-31		

The structure of the buffer used to indicate the VLAN to remove is as follows:

Table 7-124. Delete Mirror Rule Command Buffer

Byte	Description
0-1	Mirrored VLAN ID to remove. 15:12 = Reserved. 11:0 = VLAN ID.

Table 7-125 lists the delete mirror rule response (with no buffer).

Table 7-125. Delete Mirror Rule Response (Opcode: 0x0261)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0261	Command opcode.
Datalen	4-5		Length of buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: ENOENT - if SEID does not point to a valid switch element. EINVAL - if the rule ID doesn't exist. EACCES - if the VEB is not owned by this PF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-19		Reserved.
Mirror Rules Used	20-21		Number of mirror rules used by this function.
Mirror Rules Not Allocated	22-23		Total number of mirror rules still not allocated.
Data Address High	24-27		Address of buffer.
Data Address Low	28-31		



7.5 Interrupts

7.5.1 Interrupt Signaling

The X710-TM4 supports the following interrupt signaling according to per PF setting options:

- Legacy INTA/INTB/INTC/INTD interrupt messages on PCIe are supported for PFs. The X710-TM4 exposes legacy interrupt support in the *Interrupt Pin* field in the PCI configuration space of the PF. The *Interrupt Pin* parameter is loaded from the non-volatile memory (NVM) to the PFPCI_CNF register, defining the interrupt pin (A,B,C,D or none) per PF. The NVM programmer is responsible for following the PCI rules for allocating orderly interrupt pins for the PCI functions. The legacy interrupt is triggered by the interrupt vector zero of the PF.
- Message-Signaled Interrupt (MSI) is exposed in the MSI capability structure in the PCI configuration space of each PF. The MSI interrupt is triggered by the interrupt vector zero of the PF. The MSI capability is enabled per PF by the *MSI_En* bit in the PFPCI_CNF (loaded from the NVM).
- MSI-X enables multiple interrupts for the PFs as well as the VFs. MSI-X is exposed in the MSI-X capability structure in the PCI configuration space of all PFs and VFs. The number of supported MSI-X vectors is defined by the table size parameters in the MSI-X capability structure in the PCI configuration space of the PFs and the VFs. The table size parameters are loaded from global *MSI_X_PF_N* and *MSI_X_VF_N* parameters in the GLPCI_CNF2 register (loaded from the NVM), shared for all PFs and all VFs. The *MSI_X_PF_N* and *MSI_X_VF_N* parameters must be lower than or equal to the maximum number of MSI-X vectors per function as described in the text that follows. The X710-TM4 supports up to 1168 MSI-X vectors that are allocated to PFs and VFs uniformly as a function of the number of enabled PFs and the number of enabled VFs. The maximum number of MSI-X vectors per function is listed in [Table 7-126](#).

Table 7-126. MSI-X Vector Allocation per Function

Index of Maximum Enabled PF	Max Number of MSI-X Vectors per PF	Number of Registers per PF Indicated as INTPF	Index of Max Enabled VF	Max Number of MSI-X Vectors per VF	Number of Registers per VF Indicated as INTVP
1 ... 4	1 + 128	128	1 ... 32	1 + 16	16
5 ... 8	1 + 64	64	33 ... 64	1 + 8	8
9 ... 16	1 + 32	32	65 ... 128	1 + 4	4

Note: The index of the maximum enabled PF and the index of the maximum enabled VF are defined by the values of the *PCIPFCNT* and *PCIVFCNT* fields, respectively in the GLGEN_PCIFCNCNT register (loaded from the NVM).

All registers with attribute INTPF and INTVP are reflected in the function’s space as 512 registers, which is the total number of these registers in the device. Still each function might access only its own registers (starting at register index zero) according to the number of registers listed in [Table 7-126](#).

7.5.1.1 INTVP and ITVF Register Mapping

The INTVP registers are mapped in the PF space as follows:



The register index in the PF space for register 'n' of VF 'm' = number of registers per VF * 'm' + 'n', while 'm' is the relative VF index within the PF and the number of registers per VF can be 4, 8 or 16, listed in [Table 7-126](#).

The VP registers are mapped in the PF space as follows:

The register index in the PF space for VF 'm' = 'm', while 'm' is defined the same as described in [Section 7.5.1.1](#).

A register address in the PF space = base address of the specific registers + register index * 4.

7.5.1.2 Interrupt Enable Procedure

Interrupts are enabled at three levels:

- Enablement on the PCIe interface by PCI configuration registers programmed by the operating system.
 - Legacy INTA/INTB/INTC/INTD interrupt message is controlled by the *Interrupt Disable* flag in the Command register in the PCI config space per PF.
 - MSI is enabled by the *MSI Enable* flag in the MSI capability structure in the PCI configuration space per PF.
 - MSI-X is enabled by the *MSI-X Enable* flag in the MSI-X capability structure in the PCI configuration space per PF and per VF and further enablement by the *Mask* bit per MSI-X vector in the MSI-X table structure.
- Enablement of interrupts by the software device driver using the *INTENA* flag in the xxINT_DYN_CTL0 and xxINT_DYN_CTLN registers (where xx=PF or VF).
 - The software device driver sets the *INTENA* flag and clears *WB_ON_ITR* to enable the relevant interrupt signal. Upon interrupt assertion, the *INTENA* flag and (in case of legacy interrupts), the interrupt level are auto-cleared. Auto-clearing *INTENA* can be disabled globally by *GLINT_CTL*.
- Enablement of interrupts per cause using the *CAUSE_ENA* flag in the cause control registers (see [Section 7.5.2](#)).
- Interrupt moderation:
 - Interrupt throttling (ITR) is described in [Section 7.5.4.1](#)
 - Interrupt rate limiting (INTRL) is described in [Section 7.5.4.2](#)

7.5.1.3 Pending Bit Array (PBA)

In addition to the interrupt signaling on the PCIe bus, the X710-TM4 also supports the standard PBA structure in the MSI-X BAR (see [Section 11.2.6.1](#)). The PBA is relevant only when MSI-X is enabled. It is described as part of the MSI-X capability structure in [Section 11.3.3](#) for the PF and [Section 11.5.3.1](#) for the VFs. A bit in the PBA is set to one when an interrupt is triggered internally and cleared when the MSI-X vector is sent on the PCIe bus.



7.5.1.4 Interrupt Sequence

This section describes the interrupt sequence of events starting when an internal event triggers an interrupt and ending when it is sent to the PCIe bus, and also describes the expected software response. Note that the description of the interrupt sequence refers to flags that are explained in the following sections.

MSI and MSI-X Interrupts While Interrupts are Enabled

1. Any of the interrupt causes has an event that sets an internal *INTEVENT* flag for the matched interrupt signal.
2. If the interrupt is enabled by *INTENA* and also enabled by the interrupt moderation policy (ITR and rate limiting), hardware executes the steps in the following order:
 - Scan all queues associated with this interrupt by a linked list explained in [Section 7.5.3](#). Write back the status of all completed descriptors that were not reported so far and clear the internal *EVENT* flags of these queues.
 - Set the matched bit in the pending interrupt block array (PBA). Note that the PBA is reflected externally only with MSI-X (and not with legacy interrupts nor with MSI).
 - *INTEVENT* and *INTENA* are auto-cleared (auto-clearing *INTENA* is enabled by the *GLINT_CTL* register).
3. If the interrupt is enabled by the operating system (by PCIe setting), the interrupt message is sent to PCIe.
 - The interrupt indication in the PBA is auto-cleared.
4. During interrupt handling, software processes each individual interrupt cause. Specific to interrupt zero of the function, the other cause's interrupt software can optionally read the *ICR0* register, and identify the interrupt causes to be processed. Reading the *ICR0* register clears it, making it ready to reflect the events of the next interrupt.
5. At the end of interrupt handling, software re-enables the interrupts by setting *INTENA*.
 - On the same register, software also sets the *CLEARPBA* flag that clears the matched bit in the PBA.
 - On the same register, software can update one of three ITRs of this interrupt by setting the *ITR_INDX* and *INTERVAL* fields. Setting *ITR_INDX* to 11b does not impact the ITRs. See ITR explanation in [Section 7.5.4.1](#).

MSI/MSI-X Interrupts While Interrupts are Disabled by the Operating System

Steps 1 and 2 are the same as described in [Section 7.5.4.2](#).

3. The operating system polls the PBA and schedules the interrupt handler.

Steps 4 and 5 are the same as described in [Section 7.5.4.2](#).

Legacy Interrupts While Interrupts are Enabled by the Operating System (Mapped to Interrupt Zero of the PF)

Steps 1 and 2 are the same as described in [Section 7.5.4.2](#).

3. If the interrupt is enabled by the operating system (by PCIe setting), the interrupt message is sent to the PCIe.
4. At the very beginning of the interrupt service routine, the software device driver clears the internal PBA by setting the *CLEARPBA* flag in the relevant *PFINT_DYN_CTL0* register.
 - As a result, an interrupt de-assertion message is sent on the PCIe bus.



- Note:** When several PFs are assigned to the same legacy interrupt, the interrupt de-assertion message must be sent only when all PFs associated with that interrupt have cleared the internal PBA, as described in [Section 7.5.1.4](#).
- The software device driver also reads the PFINT_ICR0, scheduling the matched processes for the active flags in the register.
5. During interrupt handling, software processes each individual interrupt causes. Software can optionally read the ICR0 register, identifying the interrupt causes to be processed. Reading the ICR0 register clears it, making it ready to reflect the events of the next interrupt.
 6. At the end of interrupt handling, software re-enables the interrupts by setting *INTENA*.
 - On the same register, software can update one of three ITRs described in [Section 7.5.1.4](#).

7.5.2 Interrupt Causes

This section lists all interrupts causes (sources) while mapping these causes to the interrupt vectors described in [Section 7.5.3](#). Note that only the PF has access to any of the cause registers listed in this subsection. Therefore, the VF must request its cause-mapping programming from the PF by admin command or any other sideband channel out of scope of this document.

7.5.2.1 LAN Transmit Queues

The X710-TM4 supports 1536 LAN transmit queues for the entire device while each queue is a potential interrupt cause. A status reporting of a completed descriptor with the *EOP* bit set is considered as a transmit event that can trigger an interrupt (if the interrupt is enabled).

LAN transmit queue 'n' is enabled for interrupts using the *CAUSE_ENA* flag in its QINT_TQCTL[n] register. The queues are mapped to any interrupt vector within the function space by the *MSIX_INDX* field in the QINT_TQCTL[n] register, and are mapped to any of its ITRs (or immediate interrupt) by the ITR_INDX in this register. See the ITR description in [Section 7.5.4.1](#). Using interrupt vector zero, the transmit queues are mapped by the *MSIX0_INDX* field in the QINT_TQCTL[n] register to one of the *QUEUE_x* flags in the xxINT_ICR0 registers. The *MSIX0_INDX* of PF queues can be set to 0...7. VF queues can only be set to 0...3.

7.5.2.2 LAN Receive Queues

The X710-TM4 supports 1536 LAN receive queues for the entire device while each queue is a potential interrupt cause. A DMA completion of a descriptor with an *EOP* flag and its buffer is considered a receive event that triggers an interrupt (if the interrupt is enabled). Furthermore, if the number of free descriptors on the receive queue drops below the threshold, it is considered an immediate event (described in the following subsections). The low threshold is defined by the *LRXQTRESH* parameter in the receive queue context.

As with LAN transmit queues, the LAN receive queues are mapped to any ITR of any interrupt vector by the matched QINT_RQCTL registers.

- Note:** Software might dynamically switch between *WB_ON_ITR* mode and ITR mode by setting/clearing the *WB_ON_ITR* flag.



7.5.2.3 Other Interrupt Causes

The X710-TM4 supports asynchronous events that can generate interrupts for the PFs and the VFs. The other interrupt events supported by the PFs are indicated in the PFINT_ICR0 register, and are enabled by the PFINT_ICR0_ENA register (per PF) as listed in [Table 7-127](#). The other interrupt events supported by the VFs are indicated in the VFINT_ICR0 register, and are enabled by the VFINT_ICR0_ENA register (per VF) as listed in [Table 7-128](#).

The other interrupt cause is mapped to MSI-X vector zero of the functions. It is mapped to any of its ITRs (or immediate interrupt) as programmed by the *OTHER_ITR_IND*X field in the PFINT_STAT_CTL0 register for the PF and the VFINT_STAT_CTL0 register for the VF. (See the ITR description in [Section 7.5.4.1](#)).

During normal operation, xxINT_ICR0 is used only as a read/clear register by software. Setting the flags in the xxINT_ICR0 register (other than the queue flags and the *SWINT* flag), emulates an interrupt event of the specific cause (if enabled by the xxINT_ICR0_ENA register).

Table 7-127. Other Interrupt Causes of the PFs

PF Other Cause	Description
ECC_ERR	Unrecoverable ECC Error. This bit is set when an unrecoverable error is detected in one of the device memories.
MAL_DETECT	Malicious programming detected.
TIMESYNC	Any of the TimeSync interrupt causes as described in Section 8.5.7 .
GRST	Global Resets Requested (CORER, GLOBR or EMPR).
VFLR	VFLR was initiated by one of the VFs of the PF. The PF should read the GLGEN_VFLRSTAT getting an indication for the VF that generated the VFLR.
GPIO	GPIO event indicates an event on any of the GPIO pins enabled for interrupt by the PFINT_GPIO_ENA register. The GPIO state can be fetched on the GLGEN_GPIO_STAT register. The level transition that generates an interrupt is set for GPIO 'n' by the <i>INT_MODE</i> field in the matched GLGEN_GPIO_CTL[n] register.
HMC_ERR	HMC error as indicated in the PFHMC_ERRORINFO and PFHMC_ERRORDATA registers.
ADMINQ	Send/Receive Admin queue interrupt indication.
SWINT	Software interrupt (detailed in Section 7.5.2.4).

Table 7-128. Other Interrupt Causes of the VFs

PF Other Cause	Description
ADMINQ	Send / Receive admin queues interrupt.
SWINT	Software interrupt (detailed in Section 7.5.2.4).

7.5.2.4 Software Initiated Interrupt

In some cases, software might not be able to process all events in a single interrupt handler. In such cases, software might schedule another interrupt to complete processing all interrupt events. Software can trigger an interrupt on any vector and on any of its ITRs or NoITR. A software interrupt is mapped



to one of three ITRs or immediate interrupt by the *SW_ITR_INDX* field in the *xxINT_DYN_CTLx* registers ('xx' stands for PF or VF and 'x' stands for '0' or 'N'). When programming the *SW_ITR_INDX* parameter, the *SW_ITR_INDX_ENA* flag in this register should be set as well.

Software initiates the interrupt by setting the *SWINT_TRIG* flag in the *xxINT_DYN_CTLx* registers.

Software has the option of setting the *SWINT_TRIG* flag in the *xxINT_DYN_CTLx* registers with or without setting *INTENA* by using the *INTENA_MSK* flag in the same register. Setting or clearing the *INTENA* flag, not as part of an interrupt handling routine, might lead to race conditions and therefore it is expected that software never clears the *INTENA* flag and clearing the *INTENA* flag is always done by hardware for non-debug conditions. Also, it is expected that software sets *SWINT_TRIG* together with *INTENA* only as part of the interrupt handling routine (such as at the end of handling an interrupt).

Setting the software interrupt in vector 0 of the PFs, its status indication is reflected in the *PFINT_ICR0* register (on top of the interrupt triggering).

7.5.2.5 Interrupt Status Registers

During normal operation, software avoids any possible read accesses. Interrupt zero is used for all the other causes that enforce read cycles. The X710-TM4 provides a status indication of all causes of interrupt zero of the PFs and the VFs. The *xxINT_ICR0* registers ('xx' stands for PF or VF) provide indication for the following events:

- Any of the other interrupt causes.
- Up to 8 status flags (*QUEUE_0* ... *QUEUE_7*) in the PFs and up to 4 status flags (*QUEUE_0* ... *QUEUE_3*) in the VFs for any LAN transmit or receive queues associated with interrupt zero.
- *SWINT* indication that is a result of a software initiated interrupt.

The status indication in the *xxINT_ICR0* registers is independent of the *xxINT_ICR0_ENA* registers setting. The *xxINT_ICR0_ENA* registers enable interrupt assertion by each of the *ICR0* causes. Interrupt on the PCIe is further enabled by the interrupt vector 0 logic (legacy interrupt or MSI or MSI-X 0).

7.5.3 Interrupt Linked List

The queues are linked to a specific interrupt vector by the *MSIX_INDX* field in the *xxxQCTL* registers. Specifically, the LAN queues are associated to an interrupt vector by the following registers: *QINT_RQCTL* and *QINT_TQCTL*. These queues are chained together in a linked list that is configured by these *xxxQCTL* registers.

When an interrupt sequence is triggered, X710-TM4 processes all queues in the linked list of this interrupt. While processing the queues in the linked list, hardware writes back the status for those completed descriptors that were not reported already:

- LAN receive queues. Hardware triggers a status write back of all completed descriptors.
- LAN transmit queues. Hardware writes back the completion indication of the last completed transmit descriptor that has an *EOP* flag (regardless of the *RS* bit) and was not reported already.

Note: The *xxxQCTL* registers are accessible only to the PF. A VF can assign causes to interrupts with the help of the PF using an admin command or any other sideband message. These messages are out of scope for this document.



Note: The linked list could contain LAN transmit and receive queues. In case multiple types of interrupt causes are mapped to the same interrupt, for optimized performance, they should be interleaved in the linked list if possible.

The beginning of the linked list is indicated per interrupt signal by the following parameters in the xxINT_LNKLSTx registers (while 'xx' stands for PF or VP and 'x' stands for '0' or 'N'):

- **FIRSTQ_TYPE.** The type of the first cause in the list assigned to the interrupt signal. The type can be one of the following: receive queues or transmit queues.
- **FIRSTQ_INDX.** The queue index of the first cause in the linked list assigned to this interrupt signal. For receive and transmit queues, it is the index within the PF space (both PF and its VF queues are defined relative to the PF queue space). The **FIRSTQ_INDX** could be NULL pointer for no linked list.

An interrupt vector assigned only to the other causes does not have a linked list. In this case, software should set the **FIRSTQ_INDX** to a NULL value (0x7FF).

The interrupt causes are linked to each other by similar parameters as previously described in the following cause control registers: **QINT_RQCTL**; **QINT_TQCTL** registers ('xx' stands for PF or VP and 'x' stands for '0' or 'N').

- **NEXTQ_TYPE.** The type of the next cause in the list assigned to the same interrupt signal. It is the same definition as the **FIRSTQ_INDX** previously described.
- **NEXTQ_INDX.** The queue index of the next cause in the linked list assigned to this interrupt signal. For receive and transmit queues, it is the index within the PF space (both PF and its VF queues are defined relative to the PF queue space). The **NEXTQ_INDX** is a NULL pointer for the last cause in the linked list (equals to 0x7FF).

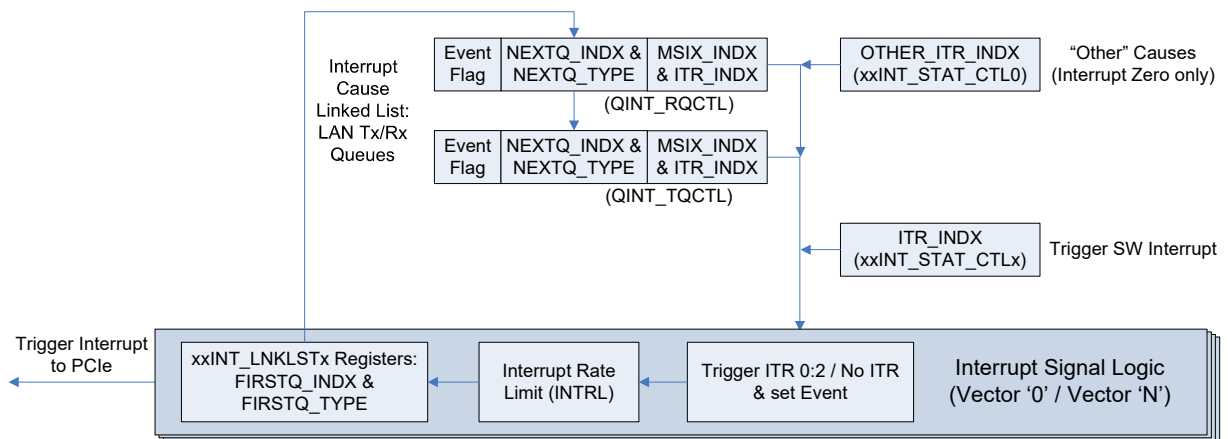


Figure 7-56. Mapping Interrupt Causes to Interrupt Signaling

7.5.3.1 Interrupt Linked List Management

This section describes the required software flow for adding and removing an interrupt cause from an interrupt linked list.

7.5.3.1.1 Initial Setting of a Linked list



Associating interrupt causes to an interrupt signal can be programmed by software by any arbitrary order. As long as interrupts are not generated, the linked list is considered static, so any programming order is acceptable.

7.5.3.1.2 Adding an Interrupt Cause to an Active Interrupt

As soon as an interrupt is active, its linked list is considered a dynamic resource. Special programming ordering is required when adding an interrupt cause as follows:

- Program the cause register as required while the NEXTQ_INDX and NEXTQ_TYPE parameters point to the next cause in the linked list.
- Update the NEXTQ_INDX and NEXTQ_TYPE parameters in the previous cause pointing to the added one.

Note: There is no need to disable the interrupt before these two steps.

7.5.3.1.3 Removing an Interrupt Cause from an Active Interrupt

Removing an interrupt cause from an interrupt linked list can be done in one of the following two cases:

Case 1. The interrupt is disabled and it is guaranteed that this interrupt does not have any possible pending interrupts. In this case, the linked list is considered static and there is no special programming ordering. Software should update the NEXTQ_INDX and NEXTQ_TYPE parameters in the previous cause to the next cause, skipping the cause that is removed from the linked list. At this point, software can modify the cause register of the removed interrupt cause if required.

Case 2. The interrupt is active. In this case, the linked list is considered a dynamic resource and software should follow a strict flow.

- Update the NEXTQ_INDX and NEXTQ_TYPE parameters in the previous cause to the next cause, skipping the cause that is removed from the linked list. In case it is the first cause in the linked list, update the FIRSTQ_INDX and FIRSTQ_TYPE parameters in the matched xxINT_LNKLSTx register, skipping the cause that is removed from the linked list.
- Clear the CAUSE_ENA flag in the matched xINT_xQCTL register.
- Software should wait a long enough time until it is guaranteed that the hardware fetched the updated value of the previous cause. Waiting a long enough time could be done by scheduling a software interrupt and waiting for the interrupt that follows.
- At that point, software can modify the cause register of the removed interrupt cause if required.

Note: There is no need to disable the interrupt before starting the previous sequence.

7.5.4 Interrupt Moderation

The X710-TM4 is able to throttle interrupts in two layered methods: Interrupt Throttling (ITR) and Interrupt Rate limiting (INTRL). These methods are detailed in the following subsections.



7.5.4.1 Interrupt Throttling (ITR)

Interrupt throttling (ITR) is a mechanism that guarantees a minimum gap between two consecutive interrupts (other than possible jitter caused by handling the interrupts). The X710-TM4 counts the time since the last interrupt is scheduled and compares it against the ITR setting. If an event associated with this ITR happens before the ITR expires, the interrupt assertion is delayed until the ITR expires. If the ITR expires before any event associated with this interrupt, the interrupt logic is armed and the interrupt can be asserted the moment the event happens. The ITR intervals per vector are programmed by the `xxINT_ITRx` registers. The ITR is measured in units of 2 μ s. Note that ITR expiration sequence is triggered only when all the following conditions are met:

- The ITR timer is expired.
- The `INTENA` or the `WB_ON_ITR` flags in the matched `GLINT_DYN_CTL` register are set.
- The interrupt has credit(s) by the interrupt rate limiting logic explained in the following section.

In addition to the terms previously mentioned, the ITR expires when the interval setting for that ITR is changed.

The X710-TM4 supports three ITRs per MSI-X vector as well as one NoITR option. The interrupt causes are mapped to one of the ITRs by the `ITR_INDX` field (per cause). The ITR intervals can be programmed directly to the `xxINT_ITRx` registers, or using the `xxINT_DYN_CTLx` registers. It might be useful to set the initial values using the `xxINT_ITRx` registers and dynamic update by the `xxINT_DYN_CTLx` registers as explained in step #4 of the interrupt sequence in [Section 7.5.1.4](#). When any ITR interval of an interrupt with pending event is expired and the `INTRL` credit is positive, (as described in the following section), the hardware follows the following steps:

- Clear the other ITRs of the same interrupt.
- Process all causes of the same interrupt (associated with all ITRs) as defined by the linked list (described in [Section 7.5.3](#)).

7.5.4.2 Interrupt Rate Limiting (INTRL)

- `INTRL` is a credit-based mechanism that limits the maximum average number of interrupts per second. The PF controls the rate limit for the vectors in its space (including its VFs) by the `PFINT_RATE0` and `PFINT_RATE_N` GL registers. `INTRL_ENA`. Enable/disable option for the `INTRL` scheme. When disabled, interrupts can be generated without rate limiting control.
- `INTERVAL`. The `INTRL` is a 6-bit interval defined in units determined by the device configuration. The value determines the time gap on which a new interrupt credit is gained.

7.5.5 Write Back on Interrupts

- Following packet reception, the status of completed descriptors is posted (write back) to host memory once every several packets or at ITR expiration. Similarly, following transmit packet DMA completion, its completed descriptors are reported to host memory as well. It is reported if the descriptors were programmed with an active `RS` flag or at ITR expiration (either descriptor write back or header write back).



NOTE: *This page intentionally left blank.*



7.6 Virtualization

7.6.1 Overview

I/O virtualization is a mechanism to share I/O resources among several consumers. For example, in a virtual system, multiple operating systems are loaded and each executes as though the whole 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 VMM by making certain functions of an I/O device shared. Thus, they can be accessed directly from each guest operating system or Virtual Machine (VM).

Two modes to support operation in a virtualized environment were implemented in previous products:

1. 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 called IOV mode in this section.
2. Central management of the networking resources by an IOVM or by the VMM (also known as software switch acceleration mode). This mode is called Next Generation VMDq mode.

The X710-TM4 fully supports Next Generation VMDq mode and SR-IOV.

X710-TM4 supports two modes of offloads as part of Next Generation VMDq: VMDq1 and VMDq2.

In VMDq1, all the VMs are part of the same VSI and each VM is allocated a single queue. There is no replication of packets to different VMs and there is no forwarding of traffic from one VMDq1 VM to another.

In VMDq2, each VM is assigned a switch port (VSI). Thus there can be full switching between ports, including VM to VM switching and replication of multicast packets.

In a virtualized environment, the X710-TM4 serves up to 384 virtual machines (VMs) per device; 128 of these can be directly assigned and any of them can be accessed in Next Generation VMDq mode. See [Section 7.6.2.4](#) for details of the VFs and VMs resource allocation.

Most configurations and resources of the device are shared across VMs. The PF driver must resolve any conflicts in configuration between the VMs. For example, the PF driver should manage all the link configuration requests of the VFs.

Most of the virtualization offload capabilities provided by the X710-TM4, apart from the replication of functions defined in the PCI-sig IOV spec, are also part of Next Generation VMDq.

A hybrid model, where some of the virtual machines are assigned a dedicated share of the port and others are serviced by an IOVM, is also supported. This model can be used when some of the VMs run OSes for which VF drivers are available. Such configurations can benefit from IOV. Others may run older OSes for which VF drivers are not available and are serviced by an intermediary. In other models, VFs are assigned to VMs requiring a higher networking bandwidth. In this last case, the IOVM or VMM is assigned some VSIs and receives all the packets with MAC addresses of the VMs behind it. VSIs are described in [Section 7.4.5.2](#).

The following section describes the support the X710-TM4 provides for virtualization. This section assumes a single-root implementation of IOV and no support for multi-root.



7.6.1.1 Direct Assignment Model

The direct assignment support in the X710-TM4 is built according to the software model defined by the SR-IOV specification.

The Physical Function (PF) driver is responsible for the initialization and handling of common port resources. Other drivers (VF drivers) might read part of the status of the common parts but cannot change it. 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 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. It is 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 this document refers to the VMM.

A channel is provided between the VF driver and the PF driver through the use of admin commands. See [Section 7.10.11](#).

7.6.1.1.1 Rationale

The purpose of direct assignment is to enable each of the virtual machines to receive and transmit packets with minimum overhead. The non time-critical operations (such as init and error handling) can be done using the PF driver. In addition, it is important that the VMs operate independently with minimal disturbance. It is also preferable that the VM interface to the 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 as follows:

1. 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.
2. Handling of the hardware ring (tail bump and head updates).
3. Interrupt handling.

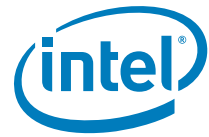
The capabilities needed to provide independence between VMs are as follows:

1. Per VM reset and enable capabilities.
2. TX QoS control.
3. Allocation of separate CSR space per VM.

The queue context creation, rate control and VF enable capabilities are controlled by the PF.

7.6.1.2 Virtualized System Overview

The following drawings describe the various elements involved in the I/O process in a virtualized system. [Figure 7-57](#) shows the flow in software Next Generation VMDq operation mode, and [Figure 7-58](#) shows the flow in IOV mode.



This document assumes that in IOV mode, the software device driver on the guest operating system is aware that it works in a virtual system (para-virtualized), and that there is a channel between each of the virtual machine drivers and the PF driver allowing message passing (such as configuration request or interrupt messages). This channel might use the mailbox implemented in the X710-TM4 or any other means provided by the VMM vendor.

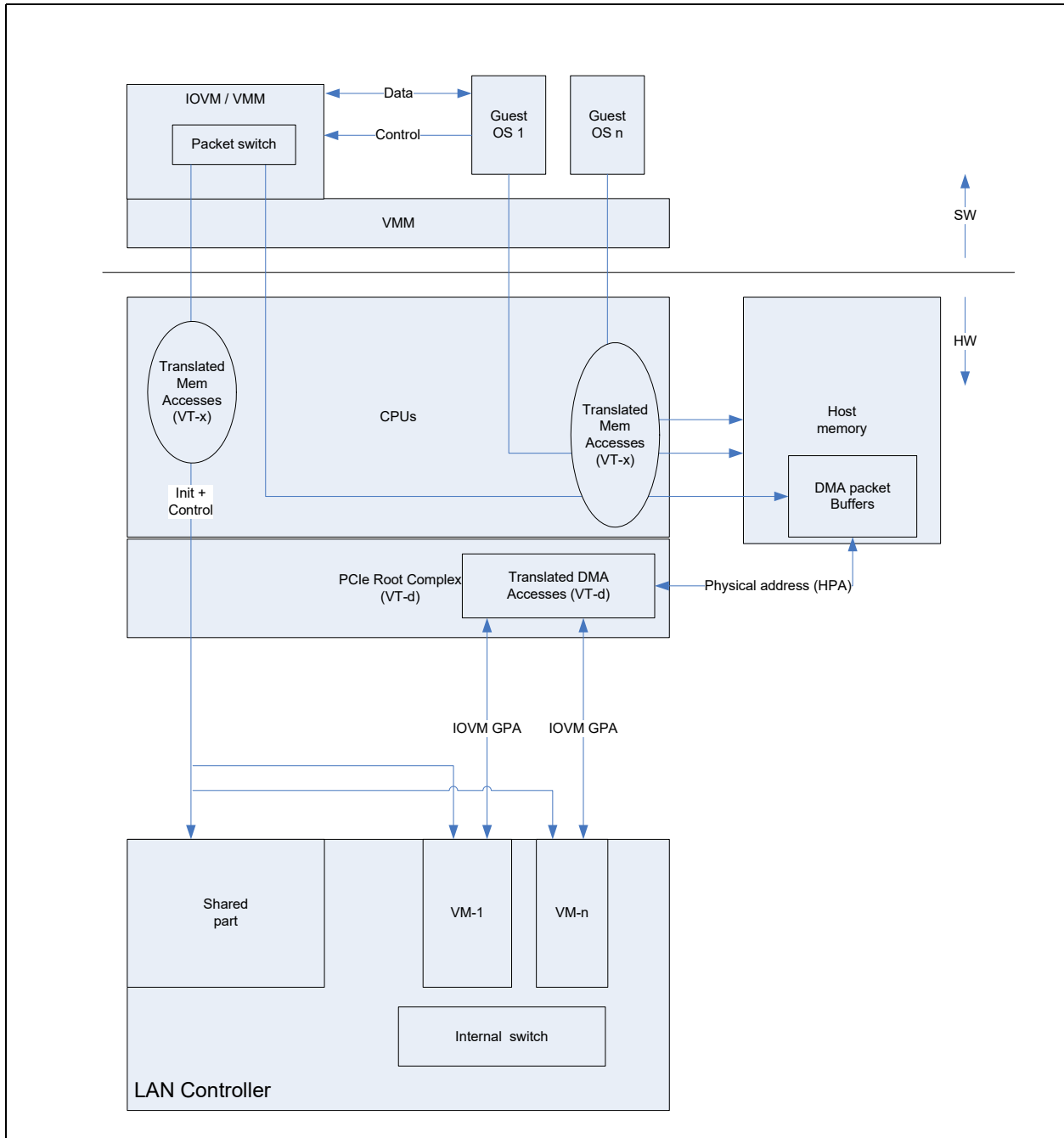


Figure 7-57. VMDq System

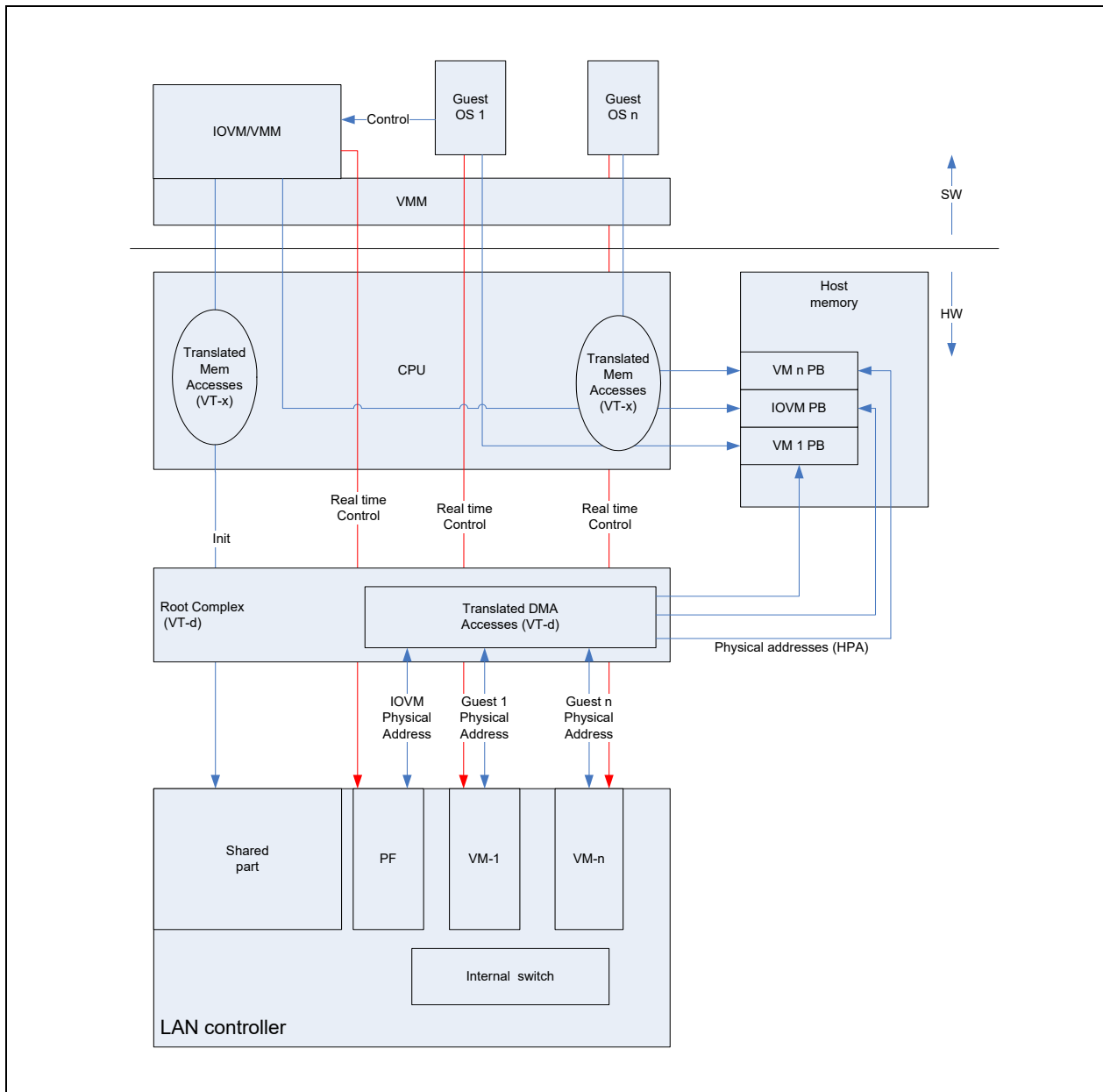


Figure 7-58. SR-IOV Based System

7.6.1.3 Virtualization Supported Features

The X710-TM4 supports a super set of the virtualization features supported in previous products. [Table 7-129](#) lists comparisons between the virtualization features of the X710-TM4 and those of the 82599.



Table 7-129. X710-TM4 Versus 82599 Virtualization Support

Feature	X710-TM4 Support	82599 Support
Direct Attach (SR-IOV) Features		
SR-IOV support	Yes	Yes
ATS support	No	No
VF to PF mailbox	Yes	Yes
Max Number of virtual functions	128 per device (globally)	64 per port (single queue)
VMDq features		
Max number of queues allocatable to VMs	1536	128
Max number of queues per VF	16	8
Max number of queues per VMDq2 VSI	16	8
Max number of queues per VMDq1 VM	1 (A VMDq1 VSI might support multiple VMs).	8
Max number of VMDq2 ports	256 per device (globally)	64 per port (single queue)
MAC addresses	1024 per device (globally)	128 per port
VLAN tags	512 per device (global)	64 per port
Queuing to pool method	SA, VLAN pairs or SA or VLAN	SA or VLAN or (SA and VLAN)
RSS per VF	Yes	No (Single RSS used for all VFs).
DCB in VF	Yes	Yes
Switching modes	VEB, VEPA, EVB	VEB
VM to VM switching	Yes	Yes
Broadcast and multicast replication	Yes	Yes
MAC and VLAN Anti spoof protection	Yes	Yes
VLAN filtering	Global and per pool	Global and per pool
Drop if no pool	Yes	Yes
per pool statistics	Yes	Yes
Per pool offloads	Yes	Yes
Mirroring	Yes	Yes
Long packet filtering	Global and per pool	Global and per pool
Storm Control	Yes	No
Promiscuous modes per VM	VLAN, multicast, unicast	Multicast

For more details about the switching support, see [Section 7.4](#).

7.6.1.3.1 Enablement of Virtualization Features

The virtualization feature is enabled using a soft SKU. [Table 7-130](#) lists the way to enable each of virtualization feature.



Table 7-130. Virtualization Features Enablement

Technology	Included Features	Enablement
SR-IOV		Set the <i>GLPCI_CAPSUP.IOV_EN</i> bit.
VEB	VEB, VEPA, VMDq2, mirroring.	Set the <i>GLGEN_STAT.VTEN</i> bit.
802.1Qbg	Port extender creation. CDCP handling.	Set the <i>GLGEN_STAT.EVBEN</i> bit.

7.6.2 SR-IOV Implementation

7.6.2.1 IOV Concepts

The SR-IOV spec defines the following entities in relation to I/O virtualization:

1. Virtual Machine (VM). A VM to which I/O resources are assigned.
2. A PCIe device. The physical device that might contain a few physical functions - in this case, the X710-TM4.
3. Physical Function (PF). A function representing a physical instance. In this case, a PCIe function that represents a physical port or a logical port. The PF driver is responsible for the configuration and management of the shared resources in the function.
4. Virtual Function (VF). A part of a PF assigned to a VM.

7.6.2.2 IOV Control

To control the IOV operation, the physical driver is provided with a set of registers and capabilities. These include:

1. The *PF_VIRT_STATUS* register: Indicates whether SR-IOV is enabled as well as the number of VFs enabled.
2. Driver to driver communication provided by the Virtualization admin commands (see [Section 7.10.11](#)).
3. Switch and filtering control admin commands (described in [Section 7.4.9](#)).
4. Reset indications and traffic enable registers per VF using the *GLGEN_VFLRSTAT.VFLRE* bit indicating that a VFLR reset occurred in one of the VFs. When the *GLGEN_VFLRSTAT.VFLRE* bit is set for a given VF, this VF can not send or receive packets. The PF should clear this bit to enable a VF.
5. Malicious driver detection (described in [Section 7.6.2.2.1](#)).

The flow used to configure a function when SR-IOV is enabled is described in [Section 4.4.1.2](#).

7.6.2.2.1 Interrupt on Misbehavior of VM (Malicious Driver Detection)

The X710-TM4 can protect itself from faulty or malicious behavior on the part of a VM driver. This is done by checking for specific illegal events.



The bits that enable these checks are grouped into three categories: Rx checks, Tx descriptor checks and Tx data checks. Individual checks are controlled by global registers GL_MDCK_RX, GL_MDCK_TCMD and GL_MDCK_TDAT respectively. If a check is not enabled (the bit is cleared) and such an event happens, the monitoring hardware does not react to the condition and the device may malfunction.

Table 7-131, Table 7-132 and Table 7-133 list the checks in each group, the register bits used to enable them, and the value read from the MDET registers when an event is detected.

The default values for these registers are loaded from NVM.

The checks apply both to VF and PF activity. If such behavior is detected, the queue is stopped and an interrupt is sent to the PF that owns the function. This means that a PF will be interrupted both on events from its queues and on events from its VFs queues. Tracking which functions have caused an event is done by reading each function’s VP_MDET_RX and VP_MDET_TX registers for VFs, or PF_MDET_RX and PF_MDET_TX for PFs. The registers are cleared by writing 1s to them. After such an event, the function needs to reset the queue. Because the malicious driver event indication in the VP/PF_MDET_TX registers is per function, and because the details of the event in the GL_MDET_TX register is common to all functions, it may not be simple to determine which queue caused the event. Therefore the driver may elect to reset the whole function.

Tx data protection is different in that it does not stop the queue; it drops the offending packet.

Two global debug registers (GL_MDET_TX and GL_MDET_RX) record the function number event ID and queue number for the first event observed on TX and RX, respectively. These registers are cleared by writing 1s.

7.6.2.2.1.1 Tx Data Checks (GL_MDCK_TDAT Register)

Table 7-131 describes the checks that are done by the X710-TM4 on data fetches for Tx data.

Table 7-131. Tx Data Checks

Check Type	GL_MDET_TDAT Bit	Description
Bad Header Length	2 MAL_LENGTH_DIS	Malicious detection of bad header length in the transmit descriptor.
Illegal Commands	3 MAL_CMD_DIS	Malicious detection of bad combination of commands in the transmit descriptor.

7.6.2.2.1.2 Tx Descriptor Validity Checks (GL_MDCK_TCMD Register)

Table 7-132 lists the checks that are done by the X710-TM4 on Tx descriptor.

Table 7-132. Tx Descriptor Validity Checks

Check Type	GL_MDCK_TCMD Bit	Event ID on MDET_TX Register	Description
Tx Descriptor Address	0 DESC_ADDR	10	Descriptor fetch failed.
Too Many Buffers	2 MAX_BUFF	14	Single send packet or large send segment is spread on more than 8 data descriptors.
Too Many Header Buffers	3 MAX_HEAD	15	LSO with header spanning more than 3 buffers.
No Header Buffers	4 NO_HEAD	24	Zero header length when large send offload is enabled.



Table 7-132. Tx Descriptor Validity Checks

Check Type	GL_MDCK_TCMD Bit	Event ID on MDET_TX Register	Description
Wrong size	5 WRONG_SIZE	16	A single send packet or large send segment that is not between min / max sizes defined in the GLTLAN_MIN_MAX_PKT register.
Tail update bigger than ring size	7 ENDLESS_TX	21	Endless transmit ring. When this flag is cleared, endless transmit ring option is enabled. When this flag is set, endless transmit ring is considered malicious event.
Bad LSO packet length	8 BAD_LSO_LEN	16	TSO Total Length not equal sum of buffers or equals to zero.
Bad LSO MSS	9 BAD_LSO_MSS	18	LSO with MSS is smaller than 64 or larger than 9668.
More than seven context descriptors	12 M_CONTEXTS	21	7 or more consecutive non-data descriptors are fetched in a transmit queue.
Bad descriptor sequence	13 BAD_DESC_SEQUENSE	18	Bad descriptors sequence. Include: MSS min/max, max number of non-data descriptors, DIF/DIX descriptors (enabled by bit 16), flex descriptors (enabled by bit 17).
Descriptor type	14 BAD_DESC_TYPE	21	Illegal descriptor type used.
No Packet	15 NO_PACKET	21	Tail update that does not contain at least one full packet.
Disable DIF / DIX descriptors	16 DIS_DIF_DIX	11	DIF/DIX descriptors are considered malicious descriptors.
Disable flexible descriptors	17 DIS_FLEX	20	Flexible descriptors are considered malicious descriptors.
Zero Bsize	18 ZERO_BSIZE	23	A descriptor with a zero BSIZE is found.

7.6.2.2.1.3 RX Checks (GL_MDCK_RX register)

Table 7-133 lists the checks that are done by the X710-TM4 on data fetches for Rx.

Table 7-133. Rx Checks

Check Type	GL_MDET_RX Bit	Event ID on MDET_RX Register	Description
RX descriptor address	0 DESC_ADDR	1	Descriptor fetch failed

The VF_NUM field in the global registers may indicate the VF, the VM or, if 0, an error in a PF owned VSI. The software device driver should deduct the actual offending agent from the reported queue number and the reporting in the per agent registers.



7.6.2.3 Hardware Resources Not Assigned to VFs

Certain device capabilities are not exposed to VFs, neither directly nor using a PF. The following features are available only to PFs or controlled solely by the PF:

- Power management and Wake-on-LAN (WoL) are PF resources and are not supported per VF.
- Link control - The link is a shared resource and as such is controllable only by the PF. This includes PHY settings, speed and duplex settings, flow control settings, etc. Flow control packets are sent using the station MAC address stored in the EEPROM. The watermarks of the flow control process and the time-out value are also controllable by the PF only. In a DCB environment, the parameters of per-TC-flow control and the ETS settings are also PF responsibilities.
- DCB policy and configuration of the device (either via DCBx or otherwise) is not open to VF control. A VF may still associate its queues with specific TCs, may request (through its PF) for Tx Scheduler nodes to be added under specific VFs, and may set the appropriate QoS fields in its transmitted packets.
- Special filtering options - Save Bad Packets is a debug feature. As such, it is available only to the PF. Bad packets are forwarded to a control VSI and thus should not be seen by a VF in regular operation.
- Reception of long packets is controlled separately per queue. Because this impacts flow control thresholds, the PF should be made aware of the decisions of all VMs. For this reason, the setup of large send packets is centralized by the PF and each VF is able to request this setting.

7.6.2.4 Hardware Resources Assignment to VFs

Table 7-134 lists how the X710-TM4 shared resources are distributed between different VFs. Details for each type of resource can be found in the relevant section of this document.

Table 7-134. VF Resource Allocation

Resource	Allocation Method	Detailed Description
General Resources		
Tx and Rx Queues	Dynamic allocation. Each VF can have a different number of queues (up to 16). Each PF allocates the queues to its VFs and their VSIs. The allocation is done using the <i>Add VSI</i> (Section 7.4.9.5.5.1) or <i>Update VSI</i> (Section 7.4.9.5.5.2) Admin commands. Queue initialization is done by the PF. Following initialization, the VF manages its queues for Tx/Rx operation.	Section 9.2
Admin Queues	An Admin Queue is allocated to each VF to communicate with its PF. The queue accesses a shared mailbox in the device.	Section 7.10
Interrupt Causes and Vectors	Allocated according to total number of exposed VFs. The interrupt causes and vectors are allocated to VFs only and not to VMs that are controlled by the PF. These VMs use the PF interrupts. 512 vectors allocated evenly among VFs Assignment of interrupt causes: Causes for Tx queues, Rx queues to interrupt vectors are done by each VF Other causes are mapped to vector 0 VFs operate with MSI-X interrupts only.	Section 7.5



Table 7-134. VF Resource Allocation (Continued)

Resource	Allocation Method	Detailed Description
Transmit Scheduling	A VF is represented in the scheduling tree through its VSI(s). Nodes associated with such a VSI may be configured with all supported attributes, such as guaranteed bandwidth, rate limiting, and arbitration scheme. Configuration is done by the PF and not directly by the VF.	Section 7.8.1
Stateless Offloads	Tx: All the regular transmit offloads like checksum and TSO are available to VFs. Enabling these offloads is done by the PF as part of the queue initialization process. Rx: All the regular receive offloads like checksum and header split are available to VFs. Enabling these offloads is done by the PF as part of the queue initialization process.	Section 8.4.4 (Tx) Section 8.3.4 (Rx)
Statistics	The VF is not directly exposed to statistics counters. If a VF needs to get statistics for its traffic, it is done via its PF.	Section 7.11
IEEE 1588	IEEE 1588 is a per link function and thus is controlled by the PF driver. VMs have access to the real time clock register via CSRs.	Section 8.5.2.2
Switching Resources		
Switch Resources	Switch configuration and resource allocation are not done by a VF directly, but through its PF. See more details in the lines below.	Section 7.4
VSI	Up to 256 VSIs can be directly supported by the X710-TM4. These VSI resources are distributed between the different PFs dynamically. Each VF is guaranteed to receive at least one VSI. The PF may allocate multiple VSIs to a VF. The allocation is done using the using the <i>Add VSI</i> (Section 7.4.9.5.5.1) admin command. There is no limit on the number of VSIs that can be assigned to a VF apart from the limit due to the number of Tx/Rx queue pairs the VF is allocated.	
Unicast MAC Addresses, Multicast Addresses, VLAN tags	The PF driver is responsible for the resource allocation to its VFs. X710-TM4 does not manage the per VF resources of the switch. The PF assigns filters to VFs VSI using the admin commands listed in Section 7.4.9.5.9 .	
Filtering Resources		
RSS	Directly controlled by each VF. RSS per VF with 64 entries and up to 16 queues.	
Flow Director	Flexible population of FD entries with VF rules. Programming is done through the PF.	



7.7 Data Center Bridging (DCB)

This section assumes the reader is familiar with the following specifications:

- IEEE P802.1Qbb-2011; Priority-based Flow Control (PFC) specification
- IEEE P802.1Qaz-2011; Enhanced Transmission Selection for Bandwidth Sharing Between Traffic Classes (ETS) specification
 - DCB Center Bridging Exchange Protocol (DCBX) specification refers to Clause 38 in the ETS specification
- IEEE Std 802.1AB-2009, Link Layer Discovery Protocol (LLDP) specification

7.7.1 Receive Path DCB

7.7.1.1 Receive Path Enhanced Transmission Selection (ETS)

7.7.1.1.1 Identifying Low Latency (LL) Traffic in Receive

Only two types of traffic are identified along the receive data path, Low Latency (LL) and Bulk (B). This approach requires a priori differentiation between low latency and bulk traffic. Traffic Class (TC) indexes are used for that purpose according to a setting made in the PRTDCB_RETSC.LLTC bitmap.

The entity that runs DCBX must configure all non-ETS TCs as LL traffic and only these TCs.

7.7.1.1.2 User Priority (UP)-to-TCs in Receive

7.7.1.1.2.1 Mapping of UP-to-TC in Receive

Register PRTDCB_RUP2TC controls the mapping of incoming packets to TCs according to the *UP* field they carry. The same mapping is used for packets received from the wires, and for those looped back internally. It defines which TC packet is stored in the Rx packet buffer and which UP bits are set in the PFC XOFF/XON frames issued to the link partner when the filling state of a TC requires it.

Packets received with the number 802.1p tag are also mapped to a TC according to the setting made in the *NOVLANUP* field of PRTDCB_RUP register that is applied as an input to the UP-to-TC mapping table of PRTDCB_RUP2TC register. To ensure that LLDP and other MAC control packets are not dropped internally by the device before they reach EMP or the host, Intel recommends that the PRTDCB_RUP.NOVLANUP field be set to the no-drop UP with the lowest index.

The *GL_SWT_L2TAGCTRL.HAS_UP* bit indicates a per-tag type if its UP is a candidate for DCB usage. The UP for DCB is taken from the first tag encountered in the packet that has this bit set. The same method is used both for receive and loopback traffic. This bit should be set for S-tags and VLAN (internal and external) EtherTypes.

7.7.1.1.2.2 Remapping the UP Field in Receive

Refer to [Section 7.2.6.2](#).



The remapping scheme applies to the traffic destined to EMP (or MC), which is represented in the tables by its own four VSI numbers.

7.7.1.1.3 Receive Flow for ETS

The X710-TM4 does not implement a true IEEE802.1Qaz standard ETS scheduler on its Rx path. Instead, it does some basic arbitration across the TCs (and UPs inside a TC) described in the sections that follow.

Receive Linked Lists

The total available PCIe bandwidth allocated for traffic reception is first allocated among the LAN ports, and secondly allocated amongst TCs (and among the different UPs attached to it) for each LAN port separately. Bandwidth allocated to a TC over a LAN port might differ to the traffic allocated to the same TC index over another LAN port. Since the number of TCs to be supported over a port might vary from 1 to 8 as per DCBX exchange, for simplicity, the Rx packet buffer is organized in a fixed manner into 32 receive linked lists, one for each UP over each LAN port.

Receive Arbitration Levels

The two levels of bandwidth allocation previously described, one across LAN ports and one across the TCs of a LAN port, induces two levels of arbitration. One Port Round Robin (PRR) arbiter that is used to select the Rx LAN port to be handled and one ETS-like arbiter that is used to select the TC to be handled within the selected LAN port. Since both arbitration schemes do not deal with absolute throughput allocation, no PCIe bandwidth should be wasted in case for some time some LAN ports or TCs offer less workload than allocated to them. Unused bandwidth by a TC is proposed first to other TCs over the LAN port, and secondly to other LAN ports.

Note: Since the host provides 2.5 GbE of single PCIe lane, the rate of arbitration decisions (configured for high bandwidth) might cause an aberration in bandwidth allocation compared to the configured one.

7.7.1.1.4 Receive ETS Arbiter

X710-TM4's receive ETS-like arbiters determine the order in which the per UP linked lists of a port are serviced at the Rx packet buffer's exit. Note that within each linked list, packets are drained in the order they arrived, irrespective of whether they arrived from the Rx port or from the Tx loopback path of the port.

Two operating modes are supported according to settings in the NON_ETS_MODE bit in PRTDCB_RETSC register:

1. Strict Priority (SP) mode - TCs are served in a strict priority order between them starting from the TC with the higher index until it has no workload, and so forth down to the TC with the lowest index.
2. Round Robin (RR) mode - TCs are served in a packet-based round-robin manner between them until no workload is left to any of them.

When several UPs are mapped (by DCBX) to the same TC index in PRTDCB_RUP2TC register, two operating modes are supported:

1. Strict Priority (SP) mode - This mode is selected by setting the UPINTC_MODE field to 0b for the corresponding TC index in PRTDCB_RETSTCC register array. UPs are served in a strict priority order between them within the total bandwidth allocated to the TC. Each time the TC is selected by the ETS arbiter, the highest UP is served first until it offers no traffic, and so forth down to the lowest UP of the TC. This is the default operating mode.



2. Round Robin (RR) mode - This mode is selected by setting the UPINTC_MODE field to 1b for the corresponding TC index in PRTDCB_RETSTCC register array. UPs are served in a packet-based round-robin manner between them within the total bandwidth allocated to the TC. Each time the TC is selected by the ETS arbiter, a different UP that offers workload is served among the UPs mapped to the TC, and so forth cyclically.

According to the total number of enabled ports and to the number of TCs of the port, these fields are configured by the DCBX agent as follow:

- Quad port configuration:
 - Ports with 1 to 4 TCs:
 - PRT_SWR_PM_THR.THRESHOLD = 0x9
 - PRTDCB_RPPMC.RX_FIFO_SIZE = 0x8
 - Ports with 5 to 8 TCs:
 - PRT_SWR_PM_THR.THRESHOLD = 0x6
 - PRTDCB_RPPMC.RX_FIFO_SIZE = 0x4
- Dual port configuration:
 - Ports with 1 to 4 TCs:
 - PRT_SWR_PM_THR.THRESHOLD = 0xF
 - PRTDCB_RPPMC.RX_FIFO_SIZE = 0x10
 - Ports with 5 to 8 TCs:
 - PRT_SWR_PM_THR.THRESHOLD = 0xC
 - PRTDCB_RPPMC.RX_FIFO_SIZE = 0x8
- Single port configuration:
 - For all ports:
 - PRT_SWR_PM_THR.THRESHOLD = 0xF
 - PRTDCB_RPPMC.RX_FIFO_SIZE = 0x10

Depending on the locality of Rx traffic, it might be that in quad port configuration, ports with 5 to 8 TCs do not achieve the Rx performance goals.

7.7.1.1.5 Rx ETS Configuration Rules

PRTDCB_RPRRC are dynamics registers that are auto-programmed by the device. The device retrieves different default values from the NVM, one for each the link speed. Each time a port changes its link speed, the corresponding default value is loaded by hardware.

PRTDCB_RUP2TC, PRTDCB_GENCL, PRTDCB_RPPMC.RX_FIFO_SIZE, and PRT_SWR_PM_THR are also dynamic registers. Refer to the flow described in [Section 7.7.3.3.1](#) for the way they are modified.

All other Rx-ETS settings described in [Section 7.5.1.1](#) are static. They must not be modified at run time. They are loaded from the NVM only at initialization/reset time.

7.7.1.2 Receive Path Priority Flow Control (PFC) and Rx Packet Buffer (RPB)

This section deals with guaranteeing no packet loss inside the X710-TM4 for the PFC-enabled traffic, whether it is received from the LAN wires or from the internal switch. Since this feature is tightly related to the allocation of receive packet buffer to the TCs, the partitioning scheme of the Rx packet buffer is described first.



RPB has a maximum byte capacity of 968 KB and a maximum packet capacity of 7744 packets (for example, 968 KB / 128 bytes).

TCs are classified in two types relatively to PFC: no-drop (such as loss less) TCs, and drop TCs (like best effort delivery). The PRTDCB_TC2PFC bitmap controls the use of PFC by a TC over a link. The setting applies for both Tx and Rx directions.

Refer to [Section 3.2.1.5](#) for the basic operation of Ethernet FC, which is relevant whether or not the device is operated in DCB mode.



7.7.1.2.1 Requirements on Rx Packet Buffer

DCB requires that separate queuing resources be allocated to each TC along the entire path from the source to the destination node. This guarantees the good functioning of the ETS scheme and mainly when it comes to prioritize LL TCs over others.

- Handle one linked list of packets per each UP, per port. The list maintains the arriving order of packets inside a flow. UPs are attached to TCs in Rx according to settings made in the PRTDCB_RUP2TC register.
- No crosstalk is sensed between the LAN ports, at least on regular basis and if the average incoming packet size is greater or equal to 128 bytes.
- Be capable to allocate fully independent buffers for up to five basic TCs per port, as follows:
 - 2 no-drop TCs and one for any other traffic type that requires PFC-enabled and 9.5 KB jumbo frames (such as iSCSI traffic).
 - 3 best effort delivery TCs (such as drop TCs) for which PFC is disabled and carry 9.5 KB jumbo frames.
- Be capable to allocate fully independent buffers for up to 8 best effort TCs per port.
- 3 extra TCs beyond the 5 basic TCs previously described, or any other TC settings up to 8 TCs per port are supported. It can however be handled via some level of buffer sharing between them.
- Partitioning the Rx packet buffer across the enabled LAN ports is set equally at init time, according to NVM settings. Even in MFP mode, no redistribution of Rx-PB occurs further to a link state change or to a link speed change. For example, all ports are assumed to operate at 10 GbE. This avoids crosstalk between ports.
- Partitioning of the amount of Rx packet buffers allocated to a port is done dynamically according to the following parameters:
 - Number of TCs supported by the port, as per DCBX.
 - Number of PFC-enabled TCs over the port, as per DCBX.
- Support LPI, and especially the worst case Tx LPI exit time of 17.38 μ s while Rx is not in LPI state.

7.7.1.2.2 Normal Partitioning of Rx Packet Buffer

- Eight dedicated pools per port, one per TC. The size of each dedicated buffer is set by the PRTRPB_DPS array of registers. Setting a null size to a dedicated packet buffer is allowed (such as for non used TCs or for the extra TCs), it is equivalent to state that no dedicated buffer is allocated to the TC.
 - Normal partitioning assumes at least the 5 basic TCs per port gets dedicated buffers.
 - Eight dedicated filling counters per port, one per TC.
 - Eight high watermarks per port, one per TC. It is set by the PRTRPB_DHW array of registers.
 - Eight low watermark per port, one per TC. It is set by the PRTRPB_DLW array of registers. At any time, it must be set to a lower value than the corresponding high watermark. Setting a null low watermark is allowed.
- One shared pool-per port. The size of the shared buffer is set by the PRTRPB_SPS register. Setting a null size to the shared buffer of a port is allowed, it is equivalent to state that no shared buffer is allocated to the port. This might be the case when only the 5 basic TCs are supported on a port.
 - One shared pool filling counter per port.
 - One high watermark per shared buffer. It is set by the PRTRPB_SHW register.
 - One low watermark per shared buffer. It is set by the PRTRPB_SLW register. At any time, it must be set to a lower value than the corresponding high watermark. Setting a null low watermark is not allowed.



- Eight occupancy counters per port, one per TC, to track the amount of bytes of the TC stored on the account of the shared buffer of the port.
- Eight high thresholds per port, one per TC. It is set by the PRTRPB_SHT register. The threshold is relative to the corresponding occupancy counter.
- Eight low thresholds per port, one per TC. It is set by the PRTRPB_SLT register. At any time, it must be set to a lower value than the corresponding high threshold. Setting a null low threshold is allowed. The threshold is relative to the corresponding occupancy counter.
- One packet buffer is allocated per port. Setting a null size to the port packet buffer is allowed (like for disabled ports), it is equivalent to state that no packet buffer is allocated to the port.
 - Normal partitioning assumes the sum of the four per-port buffer sizes equals to the total size of the Rx packet buffer in the X710-TM4, such as 968 KB.
 - The size of the per-port packet buffer is computed internally as the shared pool size plus the sum over all the dedicated pools size of the port.
 - One per port filling counter.
- One global filling counter for the entire X710-TM4's receive packet buffer.
 - One global high watermark for the entire device, relative to the global filling counter. It is set by the GLRPB_GHW register.
 - One global low watermark for the entire device, relative to the global filling counter. It is set by the GLRPB_GLW register. Normal partitioning assumes it is set to zero.

Note: The receive packet buffer accounting granularity is 16 bytes, a physical memory line. To all buffer sizes, threshold, and watermarks registers defined in bytes, the device discards the 4 least significant bits written by software.

The maximum capacity in bytes of the receive packet buffer is 968 KB, provided that the average size of the packets stored is equal or greater than 128 bytes.

- One global packet counter for the entire X710-TM4's receive packet buffer.
 - One packet high watermark for the entire device, relative to the global packet counter. It is set by the GLRPB_PHW register. It is used to reflect the implementation limit and must normally be set equal or lower than full Rx packet buffer size (like 968 KB) divided by 128 bytes.
 - One packet low watermark for the entire device, relative to the global packet counter. It is set by the GLRPB_PLW register.

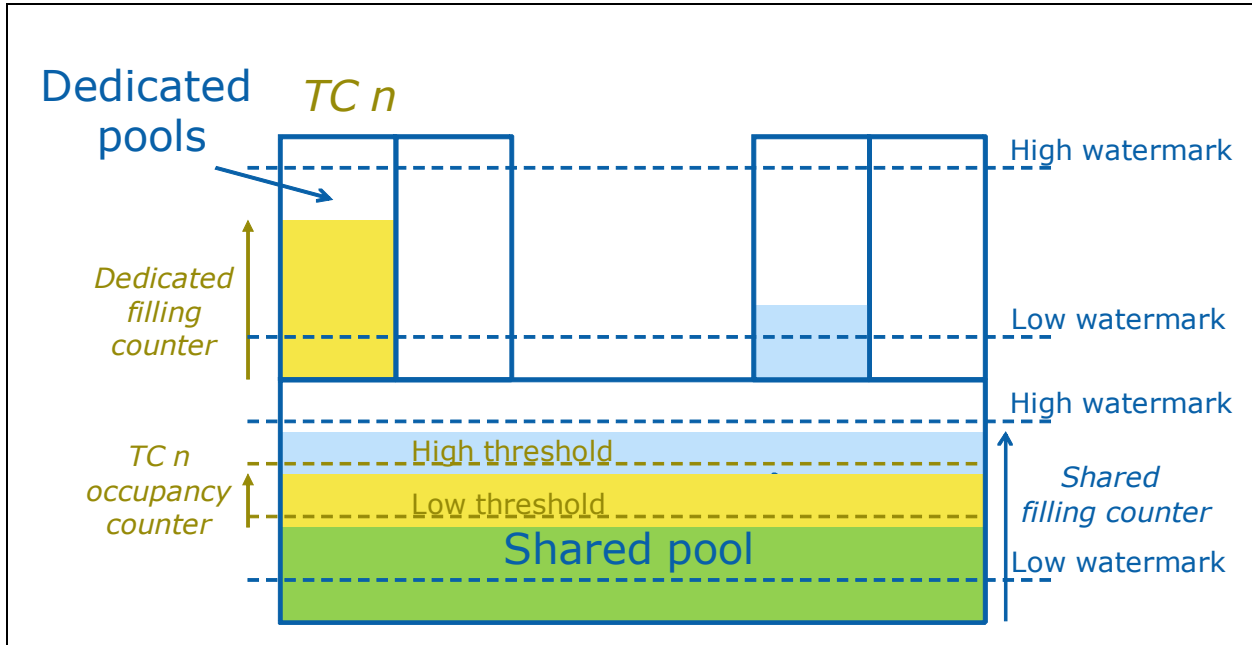


Figure 7-59. Normal Partitioning of the Per-port Rx Packet Buffer

7.7.1.2.3 Receive Drop Policy

Each time when receiving a packet attached to a drop TC, either from the LAN wire or from the internal loopback path, the packet is dropped if any of the following conditions occur when storing it into the Rx packet buffer:

- The dedicated pool (if any PRTRPB_DFC) is filled above or equal to its high watermark (PRTRPB_DHW) and the TC occupancy counter has reached its high threshold .
- The dedicated pool (if any PRTRPB_DFC) is filled above or equal to its high watermark (PRTRPB_DHW), and the shared pool filling counter has reached its high watermark and has not returned to the low watermark yet.
- The dedicated pool (if any PRTRPB_DFC) is filled above or equal to its high watermark and the global filling counter has passed above its global high watermark (GLRPB_GHW) and has not returned to the global low watermark yet.
- The global packet counter has passed above its packet high watermark.

Note: The hysteresis between high and low watermarks is required to avoid starving the no-drop TCs that can only use the shared pool account with drop TC traffic.

When receiving a packet, even for a no-drop TC, either from the LAN wire or from the internal loopback path, the packet is dropped if there would be no place for storing it entirely in the global Rx packet buffer.

7.7.1.2.4 Policy for Issuing XOFF

When receiving a packet attached to a no-drop TC, either from the LAN wire or from the internal loopback path, a PFC XOFF notification is issued onto the wire and internally to the loopback path for all the UPs mapped to the TC if any of the following conditions occur:



- a. The dedicated pool (if any PRTRPB_DFC) is filled above or equal to its high watermark (PRTRPB_DHW).
- b. No dedicated pool for the TC (PRTRPB_DPS=0) and the TC occupancy counter, (which is relative the shared pool only) has reached its high threshold (PRTRPB_SHT).
- c. No dedicated pool for the TC (PRTRPB_DPS=0) and shared pool filling counter is above its high watermark (PRTRPB_SHW). In this case, XOFF recognizes all the no-drop TCs of the port that do not have a dedicated pool.
- d. Global filling counter has reached its global high watermark (GLRPB_GHW). In this case, XOFF recognizes all the no-drop TCs of all ports.
- e. The global packet counter has passed above its packet high watermark (GLRPB_PHW). In this case, XOFF recognizes all the no-drop TCs of all ports.

The next time such a condition occurs, no XOFF is issued until the congested condition disappears (such as XON sent) or until the XOFF timer expired.

7.7.1.2.5 Accounting Packet Reception

Each time a packet is stored into the receive packet buffer (such as the packet has not been dropped), the global filling counter is increased by the number of bytes required to store the packet. Also increment the first counter for which the condition is satisfied when checking it in the following order:

1. Shared pool filling counter. Unless one of the following conditions is met:
 - a. No shared pool for the port (such as null shared pool size, PRTRPB_SPS=0).
 - b. Shared pool filling counter has reached its high watermark (PRTRPB_SHW) and there is a dedicated pool for the TC (PRTRPB_DPS>0).
 - c. TC occupancy counter (that is relative to the shared pool only) has reached its high threshold (PRTRPB_SHT) and there is a dedicated pool for the TC (PRTRPB_DPS>0).
2. Dedicated pool filling counter. Unless there is no dedicated pool for the TC (PRTRPB_DPS>0).

7.7.1.2.6 Accounting the Servicing of a Receive Packet

Each time a packet is fetched from the receive packet buffer (such as the packet is sent to the host), the global filling counter is decremented by the number of bytes used to store the packet in the receive packet buffer. Also decrement the counter(s) for which the condition is satisfied when checking them in the following order or until the amount of served bytes has been reached:

1. Dedicated pool filling counter. If both conditions are fulfilled:
 - a. There is a (non-null) dedicated buffer attached to the TC (PRTRPB_DPS>0).
 - b. The dedicated pool filling counter has still not reached zero (non-empty).

Note that it might be that the shared pool is no longer full. However, there is no hurry to send XON for a TC that has still not been served internally by Rx-ETS since it gets into XOFF state. This approach might encounter some tolerable fairness issues across the TCs.
2. Shared pool filling counter. When no or empty dedicated pool and as long as shared pool filling has not reached zero.

Note: When a port is disabled or disconnected, its Rx packet buffer is drained in such a way that all its associated filling counters get down to zero within a short and bounded time.

When a packet is served, it might be that 2 x 48 bytes = 96 bytes are not decremented from the dedicated pool filling counter and from the shared pool occupancy counter. This happens if the packet is not aligned with the 64-byte memory blocks and if packets in the TC are not served as per their insertion order.



7.7.1.2.7 Policy for Issuing XON

When servicing a packet attached to a no-drop TC (like the packet is sent to the host), a PFC XON notification is issued onto the wire and internally to the loopback path if all the following conditions are satisfied:

1. XOFF was issued for this UP.
2. TC occupancy counter is below or equal to its low threshold.
3. The global packet counter is below or equal its packet low watermark (GLRPB_PLW).
4. And the relevant condition is met:
 - a. If there is a dedicated pool for the TC and it has a non-null low watermark (PRTRPB_DLW>0) when the dedicated pool filling is below or equal its low watermark.
 - b. If there is a dedicated pool for the TC (PRTRPB_DPS>0) and it has a null low watermark (PRTRPB_DLW=0) when the shared pool filling is below or equal its low watermark.
 - c. If there is no dedicated pool for the TC. When the shared pool filling is below or equal its low watermark.

7.7.1.2.8 Prevention of PFC Crosstalk Between PCIe Functions

Though Rx queues are allocated per TC in RSS mode, in the X710-TM4, no TC index is stored in the Rx queue context. It is recommended to avoid mixing drop and no-drop traffic over the same Rx queue, as the no-drop traffic might cause head of line blocking of the drop traffic.

There is a need to prevent a malfunctioning function, a paused function, or a non-trusted VM from blocking an entire PFC-enabled TC if it does not provide Rx descriptors. For this purpose, a per UP timer is started each time a packet located at the head of the per UP linked list is waiting for software to free Rx descriptors in the Rx queue to which the packet is destined. A timer starts counting on if the UP is associated with a TC of type no drop (if the TC is of type drop, head-of-line packets are dropped when descriptors are not available).

The timer is reset when Rx descriptors are freed. The ports/UPs for which the timer timed out can be read via the GLDCB_RUPTI bitmap register. Writing 1b to a bit in the bitmap restarts the corresponding timer. The Rx queue that blocked the UP is reported to the PRTDCB_RUPTQ array of registers. The queue index reported in this register is the absolute queue index in the device space, which is different than the queue index used for the Tx and Rx queue registers.

When a timer expires, the EMP is interrupted and takes the following steps:

1. EMP identifies the port and UP of the offending queue by reading the GLDCB_RUPTI register.
2. EMP reads the PRTDCB_RUPTQ register to identify the offending queue.
3. EMP clears the timer by writing a 1b to the corresponding bit in GLDCB_RUPTI.
4. EMP reads the QTX_CTL[Q] register, where Q is the index of the offending RX queue. QTX_CTL identifies the following:

Note that this flow is based on the assumption that the software programs the QTX_CTL register for any matched transmit queues of all enabled receive queues. If this rule is not addressed by the software, the reported LAN Queue Overflow Event admin command is sent to PF0.

- a. Whether the queue belongs to a PF or VF (the PFVF_Q field).
 - A VM queue is considered to be a PF queue for this section.
 - b. The PF that owns the queue (or the parent PF in case the queue belongs to a VF) (the PF_INDX field).
 - c. The VF that owns the queue (for the case that the queue belongs to a VF) (the VFVM_INDX field)
5. If the queue belongs to a PF.
 - a. If the NVM bit PF reset on queue overflow is set.



- The EMP issues a PFR to the function. Note that the PFR also resets the PF's VFs.
 - b. If the NVM bit PF reset on queue overflow is cleared.
 - The EMP sends a LAN Queue Overflow AQ event to the PF, notifying the event.
 - c. The EMP indicates to the MC that the PF driver is not present.
6. If the queue belongs to a VF:
- a. The EMP sends a LAN Queue Overflow AQ event to the respective PF that owns the VF, notifying the event.
 - b. The PF issues a VFR to the function.

Note: It is advisable to assign manageability traffic to UPs associated with drop TCs, therefore avoiding the risk of stalling due to host queue that overflows.

7.7.1.2.9 Rx Packet Buffer Configuration Flow

All RPB registers are dynamic registers. They are loaded from the NVM at initialization/reset time and/or can be modified at run time further to a change in DCBX resolution. Re-configuring RPB while working might lead to the issuing of XOFF/XON notifications to several TCs at once.

Each time a change has been made to one of the input parameters listed in [Section 7.7.1.2.10](#), the RPB settings have to be updated.

The modified Rx-PB setting must be loaded to the device according to the following order:

1. Low thresholds and low watermarks that require to be decreased, registers PRTRPB_SLT, PRTRPB_SLW, PRTRPB_DLW.
2. High thresholds and high watermarks that require to be decreased; registers PRTRPB_SHT, PRTRPB_SHW, PRTRPB_DHW.
3. Dedicated pools size that require decreasing.
4. Shared pool size, register PRTRPB_SPS.
5. Dedicated pools size that require increasing.
6. High thresholds and high watermarks that require increasing; registers PRTRPB_SHT, PRTRPB_SHW, PRTRPB_DHW.
7. Low thresholds and low watermarks that require increasing; registers PRTRPB_SLT, PRTRPB_SLW, PRTRPB_DLW.

7.7.1.2.10 Configuration Rules for RPB Partitioning

The computing flow described in this section is run by the entity responsible for running DCBX (refer to [Section 7.5.3](#)). The flow is run for each port independently over the amount of memory allocated to the port, as part of the Rx-PB configuration flow described in [Section 7.7.1.1.2.1](#).

It makes use of the following parameters as input:

- Number of enabled ports (static parameter after initialization time), used to determine the amount of memory allocated to a port. It is set to 968 KB divided by the number of enabled ports.
- Tx LPI enabled/disabled over the port (as per PRTPM_EEER.TX_LPI_EN register bit). When Tx LPI is enabled over the port, the amount of memory usable for PFC by the port is reduced by [Tx LPI exit time - maximum MFS over all the TCs of the port]. This reduced effective packet buffer size of the port is referred to as RPB_ESize(Port). For simplicity, worst case Tx LPI exit time value is taken regardless of the port type or speed.
- Number of supported TCs in receive (PRTDCB_GENC.NUMTC).
- Number of PFC-enabled TCs (TC2PFC field in PRTDCB_TC2PFC).



- MFS per TC table (refer to [Section 7.7.3.3.3](#), it is maintained by the DCBX handling entity, and can also be loaded from NVM). MFS over a TC is referred as MFS(TC), while maximum MFS over all TCs of a port is referred as MFS(max).
- PCIe round trip time (PCIRTT).
- PFCLinkDelayAllowance (DV, for delay value) over the port. Since MFS is a per TC value, there is a different DV value per TC, referred as DV(TC).

The flow makes use of DV(TC) extended by one MFS(TC) because of a last minute packet from the same TC that can be looped back into the RPB. It is noted as $\text{Std DV(TC)} = \text{DV(TC)} + \text{MFS(TC)}$.

1. Allocate fully independent buffers to all TCs and allocate the remaining buffer space (if any) to the shared pool of the port.
 - a. Step 1 no-drop TC:
 - Dedicated no-drop low watermark = $2 \times \text{MFS(TC)} + \text{PCIRTT}$
 - Dedicated no-drop high watermark = Dedicated no-drop low watermark + $\text{MAX}\{\text{MFS(max)}, 4.5 \text{ KB}\}$; the last term is referred as MFS
 - Dedicated no-drop pool size = Dedicated no-drop high watermark + Std DV(TC)
 - Shared low threshold = 0
 - Shared high threshold = 0
 - b. Step 1 drop TC:
 - Dedicated drop low watermark = 0
 - Dedicated drop high watermark = Dedicated drop pool size
 - Dedicated drop pool size = 64 bytes
 - Shared low threshold = 0
 - Shared high threshold = $\text{RPB_ESize(Port)} - \Sigma(\text{Dedicated pools' sizes})$
 - c. Step 1 shared:
 - Shared pool size = $\text{RPB_ESize(Port)} - \Sigma(\text{Dedicated pools' sizes})$
 - Shared high watermark = Shared pool size
 - Shared low watermark = $2 \times \text{MFS(max)} + \text{PCIRTT}$
 - Exit condition: If Shared pool size $> (3 \times \text{MFS(max)} + \text{PCIRTT})$ following the space allocation for all TCs, then exit the flow

No-drop TC expected behavior for step 1 RPB partitioning:

- a. Operation:
 - An FC event in a no-drop TC must have no affect on another TC operation.
 - A drop TC operation must not affect a no-drop TC (with the possible exception of the packet occupancy cross-talk as mentioned in the text that follows).
 - b. Drop TC operation:
 - A single drop TC must be able to momentarily consume the entire shared space of the port.
 - Following a drop event in any of a port's drop TCs, the drop hysteresis mechanism must cause all drop-TCs to have the same probability of storing a packet in the RPB, thus preventing a TC from constantly blocking other TCs.
 - c. Packet occupancy cross-talk:
 - In case the packet buffer reaches its packet occupancy high watermark due to any of the TCs in any of the ports, all no-drop TCs (in all ports) must issue an XOFF event and all drop TCs (in all ports) must start dropping incoming packets until the packet occupancy is lowered to or below its low watermark.
2. Downgrade no-drop TCs to semi independent TCs. Downgrade a no-drop TC to a semi independent PB. Downgrade a no-drop TC, and so forth starting from lowest TCID and up.
 - a. Step 2 no-drop TC:
 - Dedicated no-drop low watermark = 64 bytes
 - Dedicated no-drop high watermark = $\text{MFS(TC)} + 64 \text{ bytes}$
 - Dedicated no-drop pool size = Dedicated no-drop high watermark + Std DV(TC)



- Shared no-drop low threshold = $2 \times \text{MFS}(\text{TC}) + \text{PCIRTT}$
- Shared no-drop high threshold = $2 \times \text{MFS}(\text{TC}) + \text{PCIRTT}$
- b. Step 2 shared:
 - Shared pool size = $\text{RPB_ESize}(\text{Port}) - \text{Sum of Dedicated pools' size of TCs left unchanged from Step 1} - \text{Sum of Dedicated pools' size of no-drop TCs downgraded at Step 2}$.
 - Shared low watermark = $2 \times \text{MFS}(\text{max}) + \text{PCIRTT}$ (same as in Step 1).
 - Shared high watermark = Shared pool size
 - For no-drop TCs left unchanged from Step 1: Shared low threshold = Shared high threshold = 0 (as defined in Step 1).
 - Exit condition: If Shared pool size $> (3 \times \text{MFS}(\text{max}) + \text{PCIRTT})$ following an iteration of a no-drop TC, then exit the flow.

Expected behavior for Step 2 RPB partitioning:

- a. Degraded no-drop TC operation:
 - An FC event in a no-drop TC must have no affect on another TC operation.
 - A drop TC operation might cause a no-drop TC to issue an XOFF event earlier than a non-degraded no-drop TC.
 - b. Non-degraded no-drop TC operation:
 - Upon reaching the shared high watermark due to the operation of any of the TCs in the port that have a shared space allocation (drop TCs or no-drop TCs degraded in step 2), all the drop TCs in the port must start dropping incoming packets.
 - c. Drop TC operation: unchanged from step 1.
 - d. Packet occupancy cross-talk: unchanged from step 1.
3. Downgrade TCs to shared pool only. Downgrade a no-drop TC to shared pool only, meaning it does not have a dedicated pool. Downgrade a no-drop TC, and so forth starting from lowest TCID and up.
- a. Step 3 no-drop TC:
 - No dedicated pool (dedicated pool size = dedicated high watermark = dedicated low watermark = 0).
 - Shared no-drop low threshold = $2 \times \text{MFS}(\text{TC}) + \text{PCIRTT}$
 - Shared no-drop high threshold = shared no-drop low threshold + $\text{MFS}(\text{TC})$
 - b. Step 3 shared:
 - Shared pool size = $\text{RPB_ESize}(\text{Port})$. Sum of dedicated pools' size of TCs left unchanged from step 2
 - Shared low watermark = $2 \times \text{MFS}(\text{max}) + \text{PCIRTT}$
 - Shared high watermark = Shared pool size. Max Std DV across no-drop TCs downgraded at step 3 (if any).
 - Exit condition: If shared pool size $> (3 \times \text{MFS}(\text{max}) + \text{PCIRTT} + \text{Std DV})$ following an iteration of a no-drop TC, then exit the flow.

Expected behavior for Step 3 RPB partitioning:

- a. Degraded no-drop TC operation:
 - Upon reaching the shared high watermark due to the operation of any of the TCs in the port that have a shared space allocation (drop TCs or no-drop TCs degraded in either step 2 or step 3), all the no-drop TCs in the port that were degraded in step 3 must issue an XOFF event.
- b. Non-degraded no-drop TC operation: unchanged from step 2.
- c. Drop TC operation:
 - Upon reaching the shared high watermark due to the operation of any of the TCs in the port that have a shared space allocation (drop TCs or no-drop TCs degraded in either step 2 or step 3), all the drop TCs in the port must start dropping incoming packets.
- d. Packet occupancy cross-talk: unchanged from step 1.



7.7.1.2.11 Examples of RPB Configuration Flows

- EEE enabled over all the four ports
- 9.5 KB jumbo frames used over all the TCs
- 10GBASE-T PHY is assumed, which includes the following interface delay contributors:
 - XGMII MAC/RS interface: $8\ 192 + 2 * 2\ 048 = 12\ 288$ bit times
 - 10GBASE-T delay: 25 600 bit times
- All MFS must be rounded up to 16 bytes.
- All pool sizes or watermarks must be rounded up to 64 bytes.

7.7.1.2.11.1 Example #1: 1 No-drop TC + 3 Drop TCs

Table 7-135 Example #1: 1 No-drop TC + 3 Drop TCs

10 GbE Analysis	Bit Time	KB	µs
MSS(max)	77824	9.5	7.8
PFC frame	672	0.1	0.1
Cable delay	5556	0.7	0.6
Interface delay	37888	4.6	3.8
Higher layer delay	6144	0.8	0.6
Std DV (other no-drop TCs)	327296	40.0	32.7
PCIRTT (1 µs = 10,000 bit times)	20000	2.4	1
Dedicated pool size (other no-drop TCs) at step 1	581120	70.9	58.1
Dedicated pool size (drop TCs) at step 1	176128	21.5	17.6
Number of other no-drop TCs at step 1	1		
Number of drop TCs at step 1	3		
Shared pool size at step 1	466432	57	

7.7.1.2.11.2 Example #2: 2 No-drop TCs + 3 Drop TCs

Table 7-136 Example #2: 2 No-drop TCs + 3 Drop TCs

10 GbE Analysis	Bit Time	KB	µs
MSS(max)	77824	9.5	7.8
PFC frame	672	0.1	0.1
Cable delay	5556	0.7	0.6
Interface delay	37888	4.6	3.8
Higher layer delay	6144	0.8	0.6
Std DV (other no-drop TCs)	327296	40.0	32.7
PCIRTT (1 µs = 10,000 bit times)	20000	2.4	1
Dedicated pool size (other no-drop TCs) at step 1	581120	70.9	58.1



Table 7-136 Example #2: 2 No-drop TCs + 3 Drop TCs

10 GbE Analysis	Bit Time	KB	µs
Dedicated pool size (drop TCs) at step 1	176128	21.5	17.6
Number of other no-drop TCs at step 1	2		
Number of drop TCs at step 1	3		
Shared pool size at step 1	-457216	-56	
Dedicated pool size of another no-drop TC downgraded at step 2	405504	49.5	
Dedicated pool size of a drop TC downgraded at step 2	77824	9.5	
Number of other no-drop TCs downgraded at step 2	2		
Number of drop TCs downgraded at step 2	3		
Minimum shared pool size required at step 2	406528	49.6	
Shared pool size at step 2	421683	51	

7.7.1.2.11.3 Example #3: 8 No-drop TCs

Table 7-137 Example #3: 8 No-drop TCs

10 GbE Analysis	Bit Time	KB	µs
MSS(max)	77824	9.5	7.8
PFC frame	672	0.1	0.1
Cable delay	5556	0.7	0.6
Interface delay	37888	4.6	3.8
Higher layer delay	6144	0.8	0.6
Std DV (other no-drop TCs)	327296	40.0	32.7
PCIRTT (1 µs = 10,000 bit times)	20000	2.4	1
Dedicated pool size (other no-drop TCs) at step 1	581120	70.9	58.1
Dedicated pool size (drop TCs) at step 1	176128	21.5	17.6
Number of other no-drop TCs at step 1	8		
Number of drop TCs at step 1	0		
Shared pool size at step 1	-2731520	-333	
Dedicated pool size of another no-drop TC downgraded at step 2	405504	49.5	
Dedicated pool size of a drop TC downgraded at step 2	77824	9.5	
Number of other no-drop TCs downgraded at step 2	8		
Number of drop TCs downgraded at step 2	0		
Minimum shared pool size required at step 2	720896	88.0	
Shared pool size at step 2	-1326285	-162	
Number of other no-drop TCs downgraded at step 3	6		
Number of drop TCs downgraded at step 3	0		
Minimum shared pool size required at step 3	720896	88.0	
Shared pool size at step 3	1106739	135	110.7



7.7.1.2.11.4 Example #4: 4 No-drop TCs + 2 Drop TCs

Table 7-138 Example #4: 4 No-drop TCs + 2 Drop TCs

10 GbE Analysis	Bit Time	KB	µs
MSS(max)	77824	9.5	7.8
PFC frame	672	0.1	0.1
Cable delay	5556	0.7	0.6
Interface delay	37888	4.6	3.8
Higher layer delay	6144	0.8	0.6
Std DV (other no-drop TCs)	327296	40.0	32.7
PCIRTT (1 µs = 10,000 bit times)	20000	2.4	1
Dedicated pool size (other no-drop TCs) at step 1	581120	70.9	58.1
Dedicated pool size (drop TCs) at step 1	176128	21.5	17.6
Number of other no-drop TCs at step 1	4		
Number of drop TCs at step 1	2		
Shared pool size at step 1	-1443328	-176	
Dedicated pool size of another no-drop TC downgraded at step 2	405504	49.5	
Dedicated pool size of a drop TC downgraded at step 2	77824	9.5	
Number of other no-drop TCs downgraded at step 2	4		
Number of drop TCs downgraded at step 2	2		
Minimum shared pool size required at step 2	523264	63.9	
Shared pool size at step 2	-311501	-38	
Number of other no-drop TCs downgraded at step 3	1		
Number of drop TCs downgraded at step 3	1		
Minimum shared pool size required at step 3	581120	70.9	
Shared pool size at step 3	623411	76	62.3

7.7.2 Transmit Path DCB

7.7.2.1 Transmit Path Enhanced Transmission Selection (ETS)

Transmit path ETS is handled in the Tx-scheduler. Refer to [Section 7.8](#).

This section deals with the requirements on Tx data path to avoid distortions on the ETS scheme performed by Tx-scheduler.

The approach relies on two principles:

1. Maintain the order of requests issued by the Tx-scheduler, as much as possible.



2. Avoid misleading scheduler decisions by inexact reports.

Other ETS requirements on the Tx data path:

1. Serve low latency traffic as fast as possible in any condition, as long as it does not overtake its allocated bandwidth.
2. Guarantee that best effort TCs can be served at full blown, regardless of the XOFF/XON event history over lossless TCs.
3. Achieve the per port throughput performance goals defined in [Section 14.0](#).
4. Avoid starvation of one ETS traffic class by another, even in the case where it drops packets while the other TCs offer sustained workload.

7.7.2.1.1 Identifying Low Latency Traffic in Transmit

In order to handle sudden PFC XOFF notifications received from the link, the transmit data path is provided with buffers in its different stages. To reduce impact on die size, provision is made for only two types of traffic, Low Latency (LL) and Bulk (B). This approach requires differentiating in advance between low latency and bulk traffic in order to decide in which buffer a Tx data or request for data has to be stored. TC indexes are used for that purpose according to a setting made in LLTC bitmap in the PRTDCB_TETSC_TCB and PRTDCB_TETSC_TPB registers.

The entity that runs DCBX configures all non-ETS TCs as LL traffic, and only those TCs.

7.7.2.1.2 User Priority (UP) to Traffic Classes (TC) in Transmit

7.7.2.1.2.1 Mapping of UP to TC in Transmit

Register PRTDCB_TUP2TC controls the mapping of UPs to TCs in the transmit path with respect to PFC XOFF/XON notifications received from the link partner or internally from the loopback path. It defines which transmit TC is paused/released when a UP bit is set in a PFC XOFF/XON frame received (or in internal notification).

Note: PRT_TCTUPR, PRTDCB_TUP2TC, and PRTDCB_RUP2TC registers must be programmed identically.

Tx manageability traffic issued by the MC is bound to the lowest indexed drop TC, or to TC0 if all TCs are no-drop.

7.7.2.1.2.2 Remapping of the UP Field in Transmit

Refer to [Section 7.2.6.1](#).

The remapping scheme applies also to the traffic issued by the EMP (or MC), which is represented in the tables by its own four VSI numbers.

7.7.2.2 Transmit Path Priority Flow Control (PFC)

This section deals with guaranteeing no packet loss at the link partner and inside the X710-TM4 further to receiving PFC pause frames from the link partner or further to PFC notifications received from the internal switch.



TCs are classified in two types relative to PFC: no-drop (loss less) TCs, and drop TCs (best effort delivery). PRTDCB_TC2PFC and PRTDCB_MFLCN.RPFCE bitmaps control the use of PFC by a TC over a link.

7.7.2.2.1 Performance Goals for Transmit Path PFC

7.7.2.2.1.1 Basic Assumptions

1. The X710-TM4 must support up to two PFC-enabled TCs per port. It might support more, but throughput performance would be sub-optimal performing concerning throughput.
2. One PFC-enabled TC supports 9.5 KB jumbo packets and therefore has its MFS limited to 2.2 KB.

XOFF performance goal. Once an internal/external XOFF notification is received on a TC:

1. Transmission from the TC is stopped on the wires within the Delay Value (DV) specified in Annex O of PFC specification. Refer to [Section 7.7.1.2.10](#).
2. Other TCs (PFC-enabled or not) can be served at full blown. Some limitations might be tolerated for other PFC-enabled TCs if two PFC-enabled TCs are already paused for the port, depending on the history of PFC XOFF events. Worst case is when the accumulated amount of traffic (commands or data) that was in the data path at the moment XOFF events were received on a port has reached twice the Tx command/data path depth. In this case, all PFC-enabled TCs of the port are paused.
3. PFC XOFF events received on one port must not impact performance of other PFC-enabled TCs on other ports.

7.7.2.2.1.2 XON Performance Goal

1. Over the wires. As soon as an external XON notification is received for a TC or after the XOFF timer has ended for it, resuming transmission over the wires for this TC must be made possible within the higher layer delay, plus the interface delay specified in Annex O of PFC specification. This corresponds to half of the Delay Value (DV) used for the XOFF performance goal previously described minus the cable delay.
2. On the internal switch. As soon as an internal XON notification is received for a TC, resuming transmission at the internal switch ingress port for this TC must be made possible within only the higher layer delay specified in Annex O of the PFC specification. The internal switch ingress port is located at the entry of the Rx packet buffer.

Note: Interface delay must take in account the layers present and active inside the X710-TM4 and those present in an external PHY connected to it on the board (if any).

7.7.2.2.2 Transmit PFC

The TC(s) affected by an XOFF/XON notification (which is per UP) are identified by the setting in PRTDCB_TUP2TC register.

7.7.2.2.3 PFC Dead-lock Prevention

Tx traffic that belongs to PFC-enabled TCs can be halted in the Tx pipe due to endless XOFF received either from the line or from the internal loopback path. Flushing out this traffic is a gating condition for the completion of the following operations: Tx queue disable, PFR, VFR, disabling the PFC of a TC, or DCBX UP to TC remapping.



The device implements a mechanism that automatically flushes out Tx traffic that belongs to a port for which the link goes down. However, there is no similar mechanism for flushing out traffic halted in the Tx data pipe due to endless XOFF conditions. In the case of PFR, an endless XOFF condition is detected autonomously by the device and handled as described in the sections that follow. For the other cases (previously listed), the PF(s) are responsible for detecting that the Tx data-path is halted for a long time by periodically monitoring the 8 Tx PFC timers attached to a port, one per TC (PRTDCB_TPFCTS.PFCTIMER). Each timer (per TC) is restarted by the device every time the TC is halted by an XOFF notification (received from the link). If any of these counters crosses a threshold defined by the ENDLESS_XOFF_THRESH parameter (in offset 0x15 of the EMP setting module in the NVM), the PF(s) software should take the following actions:

- Post a PFC Ignore admin command (refer to [Section 7.7.5.1](#)) for the TC, requesting the EMP to set IGNORE_FC bit(s) (32 bits - 1 per TC and per port) in the GLDCB_TFPFCI register. This causes the entire Tx data path to ignore PFC indications for the concerned TC/port, whether they were received from the line or from the internal loopback path. When in this state, the device flushes out the concerned frames without issuing them over the line or into the internal loopback path.
- When the endless XOFF condition disappears, and/or when the Tx pipe is cleaned up (read PRTDCB_TCWSTC and PRTDCB_TCMSTC arrays of registers), the PF clears the IGNORE_FC bit by posting a PFC Ignore admin command with the *Ignore* flag cleared.

Note: If the port is operating in LFC, the functionality is controlled by the bit corresponding to TC0. The 8 Tx PFC timers of a port must always be operative, even if a TC has no more UP mapped to it (for a UP-to-TC mapping change driven via DCBX).

7.7.2.3 Tx Path DCB Configuration Rules

Tx-scheduler registers are dynamic registers. They can be loaded from the NVM at initialization/reset time, and/or they can be modified at run time for a change in DCBX resolution. The order and the rules by which the registers have to be written are described in [Section 7.8.4.1](#).

PRTDCB_TUP2TC and PRTDCB_GENCL are also dynamic registers. Refer to the flow described in [Section 7.7.3.3.1](#) for instructions on modifying these registers.

All other Tx DCB settings described in [Section 7.5.2.1](#) and in [Section 7.7.2.2](#) are static. They must not be modified at run time. They are loaded from the NVM only at initialization/reset time.

7.7.3 Data Center Bridging eXchange Protocol (DCBX)

The DCBX protocol relies on the exchange with the peer of untagged LLDP packets over the physical link.

At soft SKU, where DCB is disabled (such as GLEN_STAT.DCBEN bit cleared), no DCBX nor DCB handling is permitted for any ports.

7.7.3.1 DCBX / LLDP Ownership

By default, DCBX is handled by the EMP even though when the device operates in SFP mode, the host decides to take DCBX ownership once system boot completes. SFP mode software initiates the transition on a per-port basis by posting the Stop LLDP Agent command.



Software owns DCBX; however, it might return DCBX ownership to the EMP using the Start LLDP Agent AQ command. Each time, the host that handles DCBX enters sleep mode and DCBX is uncovered (EMP does not cover for DCBX). The host is responsible for resetting the DCB configuration to its default settings (such as single TC, PFC disabled, etc.) prior to entering sleep state. When software handles DCBX, it is also responsible for configuring the DCB settings of the port by setting the DCB registers. It must use the relevant flow described in [Section 7.5.4](#). In such case, the end-station might be made by software as the DCBX master. For example, the entity that propagates its DCB settings to the peer.

When LLDP/DCB is handled by the EMP, the X710-TM4 always behaves as a DCBX slave, retrieving its DCB settings from recommendations or configurations received from the peer.

It is assumed that in MFP mode, PFs do not issue untagged LLDP frames in transmit and in receive the device is configured to capture them to the EMP. If LLDP frames are sent by the PF, they are tagged with an outer VLAN tag by the device before issuing them on the lines. In receive, LLDP frames with an outer VLAN tag are directed to the appropriate PF.

Refer to [Section 7.12](#) for the general handling of LLDP packets. The LLDP agent embedded in EMP (including its DCBX agent) is reset by GLOBR. For specific reset events, the EMP restarts the LLDP agent and DCBX resolution for the ports on which LLDP was handled by the EMP.

When LLDP is handled by the EMP following a CORER event, the EMP must reconfigure the Tx/Rx paths (and any relevant hardware that was reset) with the DCB configuration that prevailed before the reset occurred.

7.7.3.2 DCBX Version

There are two known versions of the DCBX standard: IEEE and CEE. X710-TM4 supports both and can adopt at runtime its peer's mode of operation, or it can determine the mode of operation based on NVM configuration word: DCBX mode.

The NVM configuration selects one of the three options; CEE, IEEE or adopt a peer's mode of operation.

When configured to adopt a peer's mode of operation, it is done based on DCBX TLVs received from the link peer using the flow detail that follows.

- At initialization (linkup, GLOBR or LLDP ownership hand off from software to the EMP, TLV counters expire) the EMP runs IEEE/DCBX as default mode.
- Receiving a DCBX TLVs from the partner:
 - EMP determines the DCBX mode to run. This is done based on the OUI field in the DCBX TLV. In IEEE, the OUI TLV value is 00 80 C2. In CEE, this value is 00 1B 21.
 - If the message contains IEEE/DCBX TLVs:
 - DCBX mode == IEEE
 - Ignore and CEE/DCBX TLV in the message.
 - End if
 - Else If the message contains only CEE/DCBX TLV's:
 - DCBX mode == CEE
 - End if
- Each time DCBX mode is changed between CEE and IEEE, the entire DCB configuration is re-initialized to the default DCB setting.



7.7.3.2.1 Method Used by Software to Find Which Version EMP Runs

Software sends down GET_CEE_DCBX_OPER_CFG AQ command and waits for its completion. Refer to [Section 7.12.5.2.3.9](#) for more detail on this command.

```
If retval == success
    DCB Mode = CEE.
    Negotiated DCB arbitration available in response buffer for this command.
End if

If retval == ENOENT
    DCBX Mode = IEEE.
    SW sends down "GET_LLDP_MIB" AQ command to find out the DCB arbitration negotiated using DCBX
    protocol.
End if

If retval == EPERM
    SW has taken control of DCBX. Query SW DCB agent for any DCBX configuration
End if
```

7.7.3.2.2 CEE vs. IEEE DCBX

This section highlights the main difference between CEE and IEEE modes that affect the DCBX resolution. Refer to the two standard specifications for more detail.

7.7.3.2.2.1 Operational Configuration

IEEE DCBX TLVs always advertise the operational setting of the feature. CEE DCBX on the other hand, computes the operational configuration based on local DesiredCfg and peer's DesiredCfg. If software needs to determine the OperCfg, it cannot use the GET_LLDP_MIB AQ command for CEE as OperCfg is not the wire in case of CEE. Software has to GET_CEE_DCBX_OPER_CFG AQ command for obtaining the OperCfg in CEE. Refer to [Section 7.12.5.2.3.9](#) for detail on this command.

7.7.3.2.2.2 Priority Groups

CEE DCBX has a concept of priority group that is not present in IEEE DCBX. The EMP is required to equate the priority group and TC concepts, that is, treat the priority-to-priority group mapping received in the CEE DCBX TLV as the priority-to-TC mapping as well. refer to the note in [Section 7.7.3.3.1.3](#) for more detail.

7.7.3.2.2.3 Priority Flow Control

For CEE DCBX, if the PFC doesn't become operational via the CEE DCBX state machine, then CEE DCBX disables PFC and enables link level flow control. When the EMP runs the DCBX agent in IEEE mode, this recommendation is irrelevant since it behaves in slave mode and aligns itself to its peer.



7.7.3.3 DCBX Offload Flow

DCBX TLVs embedded in the LLDP packets are identified by a TLV type value of 127 followed by the IEEE 802.1 OUI field value of 0x0080C2 for IEE and 0x001B21 for CEE.

Four types of IEEE/DCBX TLVs are handled by the X710-TM4, and they are classified into three categories as follow:

- ETS TLVs - ETS Configuration TLV and ETS Recommendation TLV.
- Priority-based Flow Control Configuration TLV.
- Application Priority Configuration TLV.

CEE/DCBX defines the following sub-TLVs:

- Control TLV - Contains fields controlling the operation of CEE DCBX.
- Priority Groups TLV - Similar to the IEEE ETS feature.
- Priority Flow Control TLV - Similar to the IEEE PFC feature.
- Application Protocol TLV - Similar to the IEEE Application Priority feature.

CEE/DCBX transmits all DCB TLVs packed inside a single LLDP TLV. IEEE/DCBX uses a separate LLDP TLV for each of the DCB features.

The generic LLDP handling flow is described in [Section 7.12](#). LLDP packets are periodically issued with DCBX TLVs inside. The following steps represent the specific handling required from the EMP based DCBX agent, upon reception of DCBX TLVs from the peer:

1. Wait for first DCBX TLV received or for a change in received DCBX TLV.
2. EMP identifies LLDP/DCBX peer - The LLDP peer identifier is made by the concatenation of its chassis ID and port ID fields. Compare these fields to a saved copy of the previous LLDP peer identifier received (if there is any and if it is not aged out).
 - a. If the LLDP peer identifier is not the same, then store the new LLDP peer identifier into the Remote DCBX parameters saved in firmware data RAM, and start a timer with the longest 802.1AB TTL value of any of the peers. Go to next step even before waiting for timer's end.
 - b. If the number of LLDP peers that run DCBX is greater than 1 at the timer's end, then a multiple peer condition is detected and reported via the PRTDCB_GENS.DCBX_STATUS field set to MULTIPLE_PEERS (as well as via a LLDP MIB Change Event if configured for). Restart the timer with the longest 802.1AB TTL value of any of the peers and exit the flow, updating the LLDP DCBX peer and treating DCBX parameters as NULL (i.e. no DCBX peer).
3. If for some reason, the desire CFG received from the peer does not match the desired CFG issued to it - CEE/DCBX Agent re-sends the DCB TLVs for the concerned feature(s) with the desired CFG copied from the peer and with the Local Parameter Change bit set.
 - Refer to the CEE DCBX specification for the way the CEE/DCBX agent handles the CEE control and feature state machines.
4. EMP resolves DCBX and reconfigures DCB. Update the RPB settings according to the flow described in [Section 7.7.1.2.10](#) and to the rules described in [Section 7.7.1.2.10](#) for the normal RPB settings. Update the local DCB management objects of [Section 7.7.3.4](#) according to the flows described in [Section 7.7.3.3.1](#) and [Section 7.7.3.3.3](#).
 - a. If LLDP TLVs were aged out, the EMP reconfigures the device with the default DCB settings, except for LFC, which should be reverted to the previously known LFC mode that prevailed before DCBX was resolved.
 - b. If some DCBX TLVs are missing or malformed, the related DCB settings are returned to their default configuration.

7.7.3.3.1 ETS TLVs Resolution

7.7.3.3.1.1 IEEE/DCBX ETS TLV's

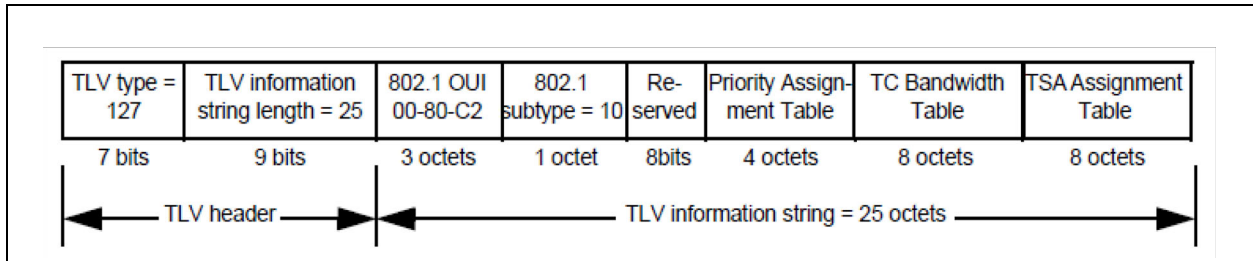


Figure 7-60. ETS Recommendation TLV Format

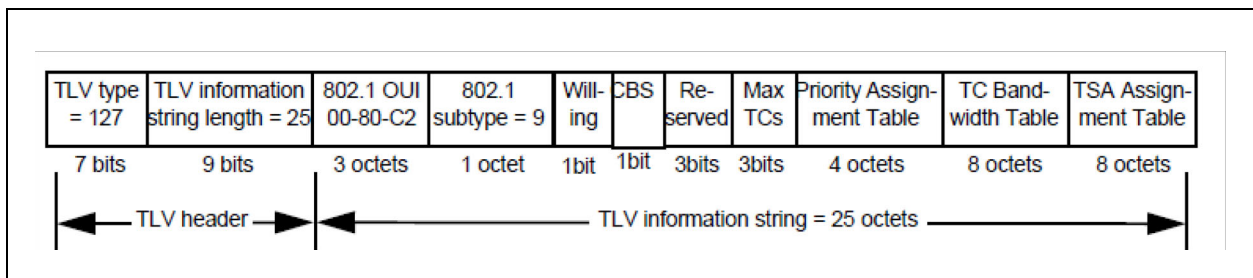


Figure 7-61. ETS Configuration TLV Format

The ETS TLV sent by a node concerns its ETS setting in transmit direction. ETS parameters resolution uses the asymmetric attribute passing state machine described in section 38.4.1 of the DCBX standard. It enables a different setting for each Tx direction between two peers. When DCBX is handled by the EMP, the X710-TM4 acts as a DCBX slave and therefore the local *Willing* bit issued to the peer is always set.

If no ETS TLV is received from the peer, or if ages out, the default ETS setting is recovered, which assumes all the UPs are mapped to a single non-ETS TC, TC 0.

Otherwise, two scenarios are possible whether or not an ETS Recommendation TLV has been received from the peer:

1. If the ETS recommendation TLV is present in peer's TLVs, local Tx ETS settings are extracted from the ETS recommendation TLV received. It is referred to as the remote ETS TLV. Local Rx ETS settings are extracted from ETS configuration TLV of the peer.
2. If no ETS recommendation TLV is present, or if it is aged out, local ETS settings (both Tx and Rx) are extracted from the ETS configuration TLV of the peer, and it is referred to as the Remote ETS TLV.

7.7.3.3.1.2 CEE/DCBX ETS TLV's

Please refer to corresponding section in CEE ETS specification.



7.7.3.3.1.3 ETS Configuration Change Flow

The EMP extracts the number of TCs from the ETS TLV. For IEEE/DCBX, it corresponds to the number of different TC indexes identified in the priority assignment table. Then it performs the following tasks sequentially, according to the case:

1. If there is a change in a PFC policy or in UP-to-TC mapping, go to the port draining flow in [Section 7.8.5.6.1.3](#).
 - a. Reconfigure DCB settings according to the following order: Load the number of TCs into PRTDCB_GENC.NUMTC.
 - b. Configure the Rx commands FIFOs as described in [Section 7.7.1.1.4](#).
 - c. For IEEE/DCBX, non-ETS TCs are those TC for which the *TSA Assignment* field value is different than two. These TCs must be marked as LL TCs in the *LLTC* field of PRTDCB_RETSC register. For CEE/DCB, there can be a unique non-ETS TC that is the TC mapped to GPID 15. This TC must be marked as LL TC in the *LLTC* field of PRTDCB_RETSC register.
 - d. Update the PFC settings to TCs according to the new priority assignment table (if it was modified), using the flow described in [Section 7.7.3.3.2](#).
 - e. Load the priority assignment table extracted from the remote ETS TLV into PRTDCB_RUP2TC.
 - f. Load the priority assignment table extracted from the remote ETS TLV into PRT_TCTUPR (refer to [Section 7.2.6.1](#)).
 - g. Mark non-ETS TCs as LL TCs in the *LLTC* field of PRTDCB_TETSC_TCB and PRTDCB_TETSC_TPB registers.
 - h. Assign MC pass-through traffic to the lowest indexed drop TC or to TC0 if all TCs are no-drop.
2. Go to the Tx-scheduler configuration flow in [Section 7.8.5.6.1.3](#), where TC bandwidth table and TSA assignment table extracted from the remote ETS TLV is loaded there into Tx-scheduler.

Note: When changing the UP-to-TC mapping and/or PFC policy, it might be that PFC and ETS behaviors become perturbed during the transition time until traffic in the pipes is emptied.

X710-TM4 always advertises support to 8 TCs, but in CEE/DCBX it locally operates with a number of TCs equal to the number of priority groups received from the peer. It is a one-to-one devices TC-to-priority group assignment made locally by the EMP. So, from TC index 0 and up, from PG index 0 and up: TC0 to PG0, TC1 to PG1 etc. PGID 15 is an exception because it is a special value dedicated to the strict priority group. For example, the priority group with unlimited bandwidth allocation. PGID 15 must be mapped to the lowest free TC index once all other priority groups have been mapped.

7.7.3.3.2 PFC Configuration TLV Resolution

7.7.3.3.2.1 IEEE/DCBX PFC TLV

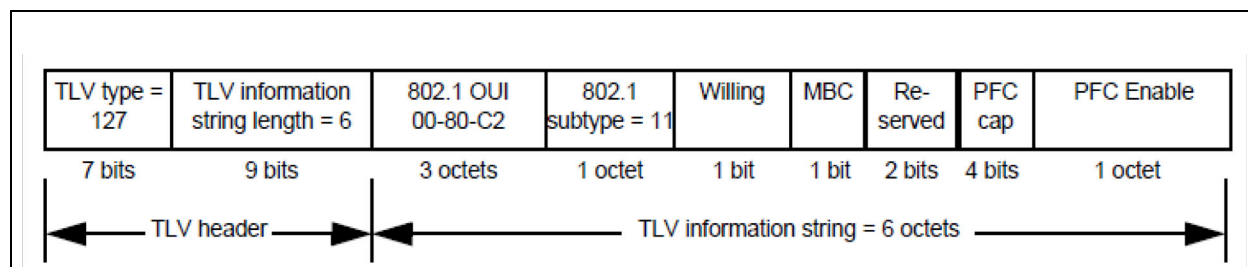


Figure 7-62. Priority-based Flow Control TLV Format



PFC parameters resolution uses the symmetric attribute passing state machine described in section 38.4.2 of DCBX standard. It means that the same setting is expected in both directions (Tx/Rx), up to the edge of the DCB network.

There are two scenarios, assuming the local *Willing* bit is always set:

1. Remote *Willing* bit is cleared. The local PFC configuration is copied from the peer.
2. Remote *Willing* bit is also set. The local PFC configuration is taken from the link partner with the lower numerical MAC address.

In scenario 1 and scenario 2, if the peer has the lower numerical MAC address, the PFC configuration change flow described in Section 7.7.3.3.2.3 is performed by the EMP:

Note: Since PFC is disabled by default on all UPs, it is recommended that the *Remote Willing* bit not be set, to make sure that the device inherits the PFC setting from the peer.

7.7.3.3.2.2 CEE/DCBX PFC TLV

Please refer to corresponding section in the CEE specification.

7.7.3.3.2.3 PFC Configuration Change Flow

- Load the *PFC Enable* bit vector of the peer into the *TC2PFC* field of PRTDCB_TC2PFC and into the *RPFCE* field in PRTDCB_MFLCN (and into PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE), making use of the PRTDCB_RUP2TC.UP2TC settings as follows:
 - a. If one of the UPs attached to a TC has its bit set in the *PFC Enable* bit vector, then set to 1b the TC2PFC bit that corresponds to the TC.
 - b. If all the UPs attached to a TC have their bit cleared in the *PFC Enable* bit vector, then clear to 0b the TC2PFC bit that corresponds to the TC.
- Set the PRTDCB_RUP.NOVLANUP field with the lowest indexed no-drop UP. If there is not a no-drop UP, use the lowest indexed drop UP instead.

7.7.3.3.3 Application Priority TLV

7.7.3.3.3.1 IEEE/DCBX Application Priority TLV

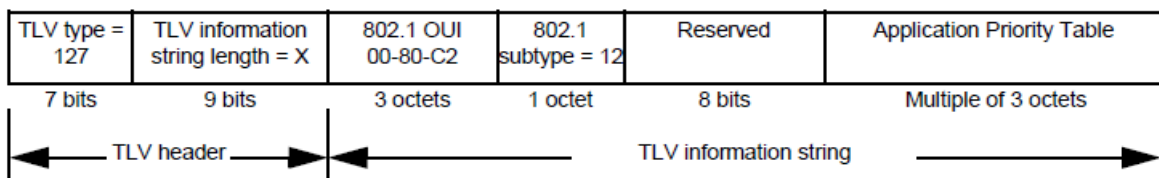


Figure 7-63. Application Priority Table TLV format

Content of this TLV is mainly informational. Two sets are stored by firmware in the DCBX database, one local set and one remote set received from the peer.

It is assumed that the peer always sends the application TLV and that the local set is copied from the remote set.



The local/remote tables stored by firmware are accessible to software via the Get LLDP MIB command.

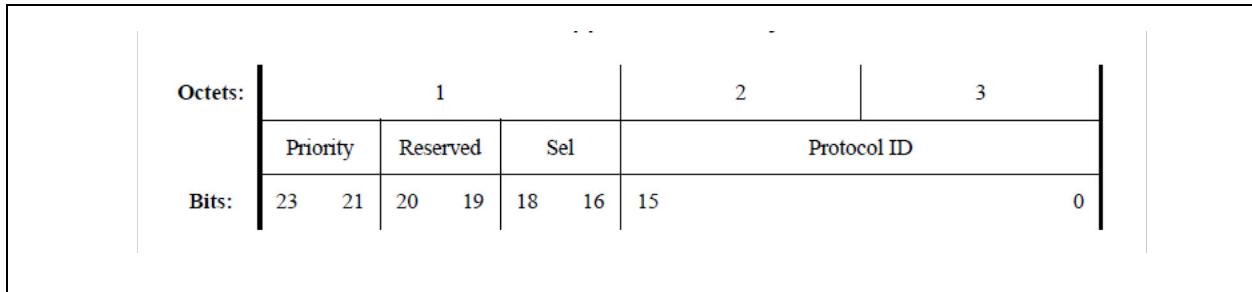


Figure 7-64. Application Priority Table

The UP attached to iSCSI traffic is the *Priority* field in the application priority table row for which:

- Sel = 2 or 4
- Protocol ID = iSCSI TCP port (3260)

7.7.3.3.2 CEE/DCBX Application Protocol TLV

Refer to the corresponding section in the CEE specification for more detail.

7.7.3.3.3 Application TLV Handling

As soon as the DCB TLVs are received from the peer (at least one TLVs), the EMP sets the *PRTDCB_GENS.DCBX_STATUS* to DONE.

If DCB TLVs age out, *PRTDCB_GENS.DCBX_STATUS* should revert back to IN_PROGRESS.

The previous flow is relevant to IEEE DCBx as well as CEE implementation.

The EMP LLDP/DCBX agent does not actively advertise iSCSI or FIP protocol entries as part of the Application TLV. But when the remote peer advertises it, the EMP mirror these protocol entries in the responded LLDPDU packets. This functionality is enabled in CEE DCBX willing mode or via IEEE DCBX (the Application TLV in IEEE does not have a willing bit). In the case of CEE DCBX, the status of these application TLV protocol entries should also be reflected in the operational configuration stored by EMP.



7.7.3.4 DCB Managed Objects

7.7.3.4.1 DCBX Managed Objects

For a DCBX port handled by the EMP, the DCBX managed objects are the DCBX parameters stored internally in the EMP data RAM and/or in device registers. All DCBX objects are per LAN port. Two instances are stored per port, one issued by the device to the link partner referred as Local DCBX parameters, and one received from the link partner referred as remote DCBX parameters.

When DCBX is handled by the EMP, the PFs must restrain themselves from any write access to the DCB registers listed in [Table 7-139](#). RW in the table refers to the EMP capability to write the object.

Table 7-139. Local DCBX Managed Objects

IEEE Object Name *CEE Object Name	Data Type	Width in Bits	Admin	Default	Related Local Configuration Register
ETS Configuration TLV					
Willing	Boolean	1	RO	1	Firmware only
Credit-based Shaper (CBS)	Boolean	1	RO	0	Firmware only
Max TCs *Num TCs Supported	Unsigned Integer	3	RW	0 ¹	PRTDCB_GENC.NUMTC
Priority Assignment Table *Priority Allocation Table	Unsigned Integer [0..7]	8x4	RW	0	PRTDCB_TUP2TC, PRT_TCTUPR, PRTDCB_RUP2TC
TC Bandwidth Table *Priority Group Allocation Table	Unsigned Integer [0..7]	8x7	RW	0	See Section 7.8.4.1
TSA Assignment Table	Unsigned Integer [0..7]	8x8	RW	0 ²	See Section 7.8.4.1
PFC Configuration TLV					
Willing	Boolean	1	RO	1	Firmware only
MBC	Boolean	1	RO	0	Hardcoded
PFC Cap *Num TC PFC Supported	Unsigned Integer	4	RO	0x8 ³	Firmware only
PFC Enable *PFC Config Table	Boolean [0..7]	8x1	RW	0	TC2PFC in PRTDCB_TC2PFC; RPFCE in PRTDCB_MFLCN; PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE; PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE; PRTMAC_HSEC_CTL_RX_ENABLE_GPP; PRTMAC_HSEC_CTL_RX_ENABLE_PPP
Application Priority Configuration TLV					
Application Priority Table *App Protocol Config Table	Unsigned Integer	32x24	RW	0	Firmware only
MFS per TC Table	Unsigned Integer	8x15 ⁴	RW	1536	Firmware only

- 0 is the encoding for 8.
- Strict priority algorithm is assumed by default (indicated by a zero value) on each user priority. Setting the value of 2 selects the ETS bandwidth allocation scheme.
- Carlsville is always able to support up to 8 PFC-enabled TCs though in some cases (when jumbo frames is enabled) fully independent PFC behavior is partially achieved.
- Maximum frame size is defined in bytes.



7.7.3.4.2 PFC Managed Objects

The PFC managed objects are PFC parameters that do not concern DCBX. RW in the following table refers to the software capability to write the object.

Table 7-140. PFC Managed Objects

Object Name	Data Type	Width in Bits	Admin	Default	Specification Reference	Related Local Configuration Register
PFC Control Objects					Table 12.1	
PFCLinkDelayAllowance ²	Unsigned Integer	16 ¹	RW	0	12.18	PRTDCB_GENC.PFCLDA
PCIRTT ²	Unsigned Integer	16	RW	0	N/A	GLDCB_GENC.PCIRTT
PFCRequests	Unsigned Integer	16	RO	0	12.18	GLPRT_PXOFFTXCNT
PFCIndications	Unsigned Integer	16	RO	0	12.18	GLPRT_PXOFFRXCNT

1. In the PFC specification the parameter is expressed in link bits, but in the X710-TM4 it is expressed in 16-byte units. SNMP agent handled in the operating system is responsible to convert the parameter in bits units when returning it in the corresponding MIB object.
2. These parameters are relevant only when DCBX is handled by the EMP. Any modification made by the PF driver to these register fields take effect on the RPB settings only once a DCB Updated admin command is posted.

7.7.3.4.2.1 PFC LinkDelayAllowance

The value of PFCLinkDelayAllowance is configurable by software per port via the PRTDCB_GENC.PFCLDA register field. It is expressed in 16-byte time units. It might be more accurate if the EMP selects by itself the PFCLinkDelayAllowance value that corresponds to the PHY, without referring to this register field.

Firmware uses this parameter to compute the Rx packet buffer settings. Refer to [Section 7.7.1.2.10](#).

Referring to Annex O in PFC Std, PFCLinkDelayAllowance is also referred as the Delay Value (DV), which is computed per TC as follows:

$$DV = 2 * (\text{Max Frame}) + (\text{PFC Frame}) + 2 * (\text{Cable Delay}) + 2 * (\text{External PHY Delay}) + (\text{Higher Layer Delay})$$

Where the additional delay of the internal PHY must be added to the Cable Delay, to get the following equation:

$$DV = 2 * (\text{Mac Frame}) + (\text{PFC Frame}) + 2 * (\text{Cable Delay} + \text{internal PHY Delay}) + 2 * (\text{Interface Delay}) + (\text{Higher Layer Delay})$$

Note: External PHY delay is TBD.

- $2 * (\text{Max Frame}) = \text{MFS (TC)} + \text{MFS (max)}$. The term is in fact formed by the sum of the MFS (TC) over the TC for which DV is computed and the maximum MFS (max) over all TCs. Refer to [Section 7.7.3.3.3](#) for the MFS per TC table to be used. The PFCLinkDelayAllowance loaded to the PFCDLA register field must assume MFS is 9.5 KB for both. Firmware is responsible to handle a different PFCLinkDelayAllowance per each TC according to the MFS per TC table it handles. It is referred as DV(TC).
- (PFC Frame) duration is equal to 672 bit times.



- 2*(Cable Delay) term is proportional to the cable length and inversely proportional to the bit time duration (like link speed). A 100m Cat6 cable operated at 10 GbE link speed is the worst case for all the supported cable length, medium type and link speeds across 10 GbE, 1 GbE, and 100 Mb/s. Under this worst case conditions the term equals to 2*(5,556) bit times. Firmware is not responsible for optimizing the term contribution if the link speed is below 10 GbE.
- 2* (Interface Delay) + (Higher Layer Delay) term must take in account the Interface Delay at each side of the link, and Higher Layer Delay at the peer side. The PFCLinkDelayAllowance value loaded to the PFCLDA register field must be computed assuming at the peer the worst case values tolerated by the standard over the medium type. Table O-1 in Annex O of PFC Std gives the Interface Delay contributors for the different layers that might be present between the controller and the physical medium.

Sublayer	Maximum RTT (bit times)	Maximum RTT (pause quanta)	Reference (subclause of 802.3)
10G MAC Control, MAC, and RS	8 192	16	46.1.4
XGXS and XAUI	2 048	4	48.5
10GBASE-X PCS	2 048	4	49.2.15
10GBASE-R PCS	3 584	7	50.3.7
LX4 PMD	512	1	53.2
CX4 PMD	512	1	54.3
Serial PMA and PMD	512	1	52.2
10GBASE-T	25 600	50	55.11

Figure 7-65. IEEE Std. 802.3-2018 Interface Delays

The (Higher Layer Delay) at 10 GbE is 614.4 ns, which is equivalent to 6,144 bit times. As previously stated, firmware is not responsible for optimizing the contribution of this term if the link speed is below 10 GbE.

7.7.3.4.2.2 PCIRTT

The value of PCIRTT is configurable by software for the entire device via the GLDCB_GENC.PCIRTT register field. It is expressed in 16-byte time units.

Firmware uses this parameter to compute the Rx packet buffer settings. Refer to [Section 7.7.1.2.10](#).

It also represents the maximum PCIe round trip time supported with no performance penalty. For example, 2 μs by default. A 10 GbE link speed is the worst case for all the supported link speeds across 10 GbE, 1 GbE, and 100 Mb/s. At 10 GbE speed, the term equals to 20,000 bit times. Firmware is not responsible for optimizing the term contribution if the link speed is below 10 GbE.



7.7.4 Initialization of the DCB Functionality

7.7.4.1 DCB Initialization Flow

DCB initialization flow is always handled by the EMP, starting from pre-boot time, for any of the following events: GLOBR, PCIR, PERST, EMPR, and POR.

1. Auto-load. The device auto-loads the default configuration of the ports from the NVM, which might include some changes to the DCB/RPB registers' defaults. In any case, DCB/RPB defaults loaded to registers at this stage must guarantee basic connectivity for the enabled ports.

It must equally partition RPB across the enabled ports by setting an identical value into their shared pool buffer (PRTRPB_SPS register) so that their sum equals 968 KB. For disabled ports, zero is expected to be loaded from the NVM into PRTRPB_SPS. DCB defaults must assume one single (non-ETS) TC for the port, like TC0, to which all UPs are mapped. No dedicated pools are used at this stage. PFC is disabled for all UPs. The control port of the internal switch must by default be addressed to the EMP. It guarantees that untagged LLDP packets are forwarded to the EMP in Rx and in Tx, and that untagged LLDP packets (wrongly) issued by the host are filtered out by the device before reaching the wire.
2. Auto-negotiation. If auto-negotiation is enabled by default, the device performs auto-negotiation with the peer, which might result in a link speed change and/or in a LFC status change. Any DCB/RPB configuration change required for auto-negotiation is handled by the device (both hardware and EMP) at pre-boot stage, as follows:
 - a. For a link speed change, update the receive port arbiter. The device updates the credits allocated to the port in the Rx port arbiter. Refer to [Section 7.7.1.1.5](#). Other link speed register settings are handled by the EMP as described in [Section 3.2.10](#).
 - b. For an LFC state change, update RPB and LFC registers. EMP computes and loads the high/low watermark of the shared pool allocated to the port as if it were a single fully independent buffer. Refer to [Section 7.7.1.2.10](#). Other LFC registers settings are handled by EMP as described in [Section 3.2.1.5](#).
 - c. For 2.5 Gb/s and 5 Gb/s speeds, follow the initialization process for 10 Gb/s link.
3. EMP runs DCBX. Once EMP has reloaded the LLDP filters in the internal switch and if LLDP agent was not disabled via NVM settings or via the Stop LLDP Agent admin command (the later case is not relevant for POR triggering events) EMP performs a DCBX exchange with the link partner from scratch and updates the device registers related to DCB managed objects as per the flows described in [Section 7.7.3.2](#).
 - a. This step starts as soon as the link is up (such as auto-negotiation completed) and any time the link comes up again later on.
 - b. For 2.5 Gb/s and 5 Gb/s speeds, the EMP firmware should:
 - Stop (do not start the DCBx agent).

Start the LLDP agent without starting the DCBx agent if firmware is the LLDP owner; otherwise, do nothing in any other case.
 - Download the NVM default values to the device registers related to DCB managed objects using the flows described in [Section 7.7.3.4](#).
 - a. In MFP mode, once the link is up, SMASH/CLP runs and determines PF to UP association.
4. PFs read the DCB configuration. PFs issue a Get LLDP MIB command to the device. From the response posted by the EMP, it identifies the UP used for storage as well as the UP-to-TC mapping used at the link level.
 - a. In MFP mode, a PF can read the UPs it is assigned by BIOS in two steps: first issue the Get Switch Configuration AQ command to get the list of VSIs assigned to it. Then it issues the Get VSI Parameters command for each assigned VSI to get the UPs assigned to the VSI.
 - b. For 2.5 Gb/s and 5 Gb/s speeds, PF software can do either of the following:
 - Issue a Get LLDP MIB command to the device and configure the software device driver.



Or

- Configure the software device driver to the default values (1 TC, 1 UP, no PFC, no LFC).
5. PFs configure VEBs and VSIs according to the flow described in [Section 7.8.5.6.1.4](#).
 - a. If several UPs are attached to a TC and if there is at least one Rx queue for each, the EMP might attach one corresponding Tx queue set per each UP under the TC. This is done upon availability of the Tx queue set resources. In this case, a Tx queue set belongs to a single UP and to a single TC. Otherwise, it belongs to a single TC. In any case, an Rx queue might not carry traffic from a single UP or a single TC, as Rx filters assign traffic to Rx queues according to many decision parameters that might not take in account only UP/TC.
 - b. EMP sets the per VSI UP translation tables used in Tx and Rx, VSI_TUPR and VSI_RUPR tables, respectively.
 - c. For 2.5 Gb/s and 5 Gb/s speeds, 1 TC and 1 UP per VSI is available. No additional configuration is required.

Note: On CORER events, EMP reconfigures the core blocks with the DCB settings saved from the last DCBX resolution or the default settings if DCBX is disabled.

7.7.4.2 Transfer of DCB Ownership Between EMP and Operating System

When link speed is 2.5 Gb/s and 5.0 Gb/s, any DCB configuration command, such as Set DCB Parameters is ignored by EMP firmware and the *I40E_AQ_RC_EMODE* response is sent by firmware. The X710-TM4 stays configured to the default values. Note that there are no changes for other link speeds.

This is done per port according to the following flow:

1. The operating system disables DCBX offload. The operating system posts a Stop LLDP Agent command. This notifies the EMP to stop handling DCBX.
 - a. In this case, the Shutdown LLDP Agent command variant is used so that the EMP LLDP agent issues a last LLDP packet to the peer with TTL=0 before returning the port to its default DCB setting (one single non-ETS TC like TC 0). This variant is used by the PF when it does not plan to take LLDP ownership to gracefully shutdown LLDP with the peer without requiring the host to get involved in the last LLDP packet transmission.
 - b. EMP keeps track of LLDP ownership per port in an internal RAM that resets only on POR events.
2. EMP completes the Stop LLDP Agent command.
3. The operating system redirects LLDP to the host. The operating system moves the LLDP forwarding rules of the internal switch to the control VSI.
4. Following each core reset, software uses the [Set DCB Parameters Command](#) to instruct the firmware to autonomously configure hardware with the default DCB settings every time a link is established.

Note: If the Stop LLDP Agent command is issued when the LLDP agent is already off, the command is silently dropped.

The LLDP agent might be disabled by default via clearing the LLDP Admin Status word in the NVM LLDP configuration module. This mode might be useful on nodes that provide NAT and other network edge services.

On any reset event other than POR, the EMP does not re-enable the LLDP agent.

The Stop LLDP Agent is ignored when the device is operating in MFP mode.

7.7.4.2.1 Transfer of DCB Ownership to the Operating System



When in SFP mode, the operating system might decide to take ownership of DCBX from the EMP. This is done per port according to the following flow:

1. The operating system disables DCBX offload. The operating system posts a Stop LLDP Agent command. This notifies the EMP to stop handling DCBX.
2. EMP completes the Stop LLDP Agent command.
3. The operating system redirects LLDP to the host. The operating system moves the LLDP forwarding rules of the internal switch to the control VSI.
4. The operating system is running DCBX. The operating system performs DCBX exchange with the link partner and updates the device registers related to DCB managed objects as per the flows described in [Section 7.7.3.2](#). It might be that the operating system reruns DCBX in master mode, propagating to the peer the local DCB settings that were made to its DCBX agent via some DCB application.
5. Each time the operating system detects a DCBX change, it posts a DCB updated event in the admin command queue to notify the EMP that the untagged traffic mapping must be modified.
6. Following every link up event, software must use the Set Local LLDP MIB command to trigger the configuration of hardware with the current DCB settings.

Note: If the Stop LLDP Agent command is issued when the LLDP agent is already off, the command is silently dropped.

The LLDP agent might be disabled by default via clearing the LLDP Admin Status word in the NVM LLDP configuration module. This mode might be useful on nodes that provide NAT and other network edge services.

On any reset event other than POR, the EMP does not retake LLDP ownership on ports where ownership has been moved to the PF. It means that the PF must not perform the transfer of DCB ownership to the operating system again further to such reset events.

7. For the link speeds 2.5 Gb/s and 5 Gb/s:
 - a. In normal flow:
 - The operating system disables DCBX offload. The operating system posts a Stop LLDP Agent command. This notifies EMP to stop handling DCBX. The EMP keeps track of LLDP ownership per port in an internal RAM that only resets on POR events.
 - The EMP completes the Stop LLDP Agent command.
 - The operating system redirects LLDP to the host. The operating system moves the LLDP forwarding rules of the internal switch to the control VSI.
 - b. For 2.5 Gb/s and 5 Gb/s:
 - The operating system doesn't runs DCBX. Any DCB configuration, except the default, should be prohibited by software device driver.

Note: Any DCBx related command, such as Set Local LLDP MIB, should be ignored by EMP firmware and the *I40E_AQ_RC_EMODE* response should be sent.

7.7.4.2.2 Return of DCB Ownership to EMP

If an operating system reboot occurs, it is possible that the system goes through a complete initialization cycle, including a BIOS pre-boot phase. In such a case, the LLDP ownership must return to the EMP so that it supports pre-boot activities such as operating system boot over network. Pre-boot software has to first initiate a GLOBR to re-initiate the device hardware, and then to signal the EMP to take over the LLDP agent. The latter is done by posting a Start LLDP Agent command. Refer to [Section 7.12.5.2.3.8](#).

In addition to receiving the Start LLDP Agent, the EMP starts the LLDP agent from scratch, making use of the hardware defaults of registers.



7.7.4.3 Initial DCB Settings

The hardware default settings for DCB registers correspond to ports are operated at 10 GbE or lower speeds.

The value put in the NVM image must be optimized to reflect the PHY type and the internal delays in the X710-TM4.

Table 7-141. Changes in DCB Registers Made via NVM Settings When Port Operated at Different Link Speed

Register	Field	1x 10 GbE	2x10 GbE	1x2.5 GbE	2x2.5 GbE	1x5 GbE	2x5 GbE	4x10 GbE
PRTDCB_GENC	PFCLDA	0x079D	0x079D					0x079D
GLDCB_GENC	PCIRTT	0x009C	0x009C					0x009C
PRTDCB_RPPMC	RX_FIFO_SIZE	0x10	0x10					0x08
PRT_SWR_PM_THR	THRESHOLD	0x0F	0x0F					0x09
GLRPB_PHW	PHW	0x1246	0x1246					0x1246
GLRPB_PLW	PLW	0x0846	0x0846					0x0846
PRTRPB_SPS	SPS	0x3C800	0x3C800					0x3C800

Table 7-142. Changes in DCB Registers Made via NVM Settings When Port Operated at Different Link Speed

Register	Field	1x10G	2x10G	4x10G
PRTDCB_GENC	PFCLDA	0x079D ¹	0x079D ¹	0x079D ¹
GLDCB_GENC	PCIRTT	0x009C	0x009C	0x009C
PRTPE_RUPM_PFCTCC	LL_PRI_EN	0x1	0x1	0x1
	LL_PRI_TH	0x9	0x9	0x9
	TCOFFTH	0xA	0xA	0xA
PRTPE_RUPM_PFCPC	PORTOFFTH	0x14	0x14	0x14
PRTPE_RUPM_THRES	MAXSPADS	0x22	0x22	0x22
	MAXSPADSPERC	0xA	0xA	0xA
GLPE_RUPM_CQPPOOL	CQPSPADS	0x8	0x8	0x8
GLPE_RUPM_PTXPOOL	PTXSPADS	0x25	0x25	0x25
GLPE_RUPM_FLRPOOL	FLRSPADS	0x8	0x8	0x8
GLPE_RUPM_PUSHPOOL	PUSHSPADS	0xA	0xA	0xA
GLPE_RUPM_GCTL	RUPM_DIS	0x0	0x0	0x0
	SWLB_MODE	0x0	0x0	0x0
PRTPE_RUPM_CTL	RUPM_MODE	0x0	0x0	0x0
GLDCB_TGENC_TUPM	TCPM_DIS	0	0	0
	CWLB_MODE	0	0	0
GLDCB_TGENC_TLPM	TFPM_DIS	0	0	0
	FWLB_MODE	0	0	0



Table 7-142. Changes in DCB Registers Made via NVM Settings When Port Operated at Different Link Speed

Register	Field	1x10G	2x10G	4x10G
PRTDCB_TCPFPCPC	PORTOFFTH	0x06C4 ²	0x065C	0x034A
PRTDCB_TCPFCTCC	TCOFFTH	0x0624 ²	0x0320 ²	0x019E
	LL_PRI_EN	0x1	0x1	0x1
	LL_PRI_TRESH	0x608	0x304	0x182
PRTDCB_TCPMC	CPM	0x0624 ²	0x0320 ²	0x019E
	TCPM_MODE	0x0	0x0	0x0
PRTDCB_TDPMC	DPM	0x36 ²	0x36 ²	0x13
	TCPM_MODE	0x0	0x0	0x0
PRTDCB_TFPFCC	PORTOFFTH	0x76 ²	0x76 ²	0x32
	TCOFFTH	0x36 ²	0x36 ²	0x13
GLSCD_LANTCBCMDS	NUMLANTCBCMDS	0x4C	0x4C	0x4C
GLDCB_RSPMC	RSPM	0x44	0x44	0x44
	RPM_MODE	0x0	0x0	0x0
	PRR_MAX_EXP	0xB	0xB	0xB
	PFCTIMER	0x800	0x800	0x800
	RPM_DIS	0x0	0x0	0x0
PRTDCB_RPPMC	LANRPPM	0x0A	0x0A	0x0A
	RDMARPPM	0x0F	0x0F	0x0F

1. The value put in the NVM image must be optimized to reflect the PHY type and the internal delays in the X710-TM4.
2. The value is 0 for the disabled ports.

7.7.5 DCB Admin Commands

All parameters in the admin commands are defined in little endian.

7.7.5.1 PFC Ignore

This command is used to request that the device ignore PFC condition present on a Tx path for a TC. The same command is used to release a PFC ignore request. When PFC packets are ignored, they are considered as regular packets and might be forwarded to host.

Table 7-143. PFC Ignore Response

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0301	Command opcode.
Datalen	4-5		Must be zeroed.



Table 7-143. PFC Ignore Response (Continued)

Name	Bytes.Bits	Value	Remarks
Return Value	6-7		Return Value: 0x0 - No error (success)
Cookie High	8-11	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Completion_flags	16		Bits 7:0 - TC Status Bitmap: Returns the TCs for which PFC is currently ignored. Bit n set to 1b means PFC condition on Tx path for TC n is ignored by the device.
Reserved	17-19		Reserved, must be zeroed.
Reserved	20-23		
Reserved	24-27		
Reserved	28-31		

Table 7-144. PFC Ignore Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0301	Command opcode.
Datalen	4-5		Must be zeroed by the software device driver.
Return Value/ VFID	6-7		Must be zeroed by the software device driver.
Cookie High	8-11	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Command_flags	16-17		Byte 16, bits 7:0 - TC bitmap: Bitmap of the TCs concerned by the command. Bit n set to 1b means TC n is concerned by the request. When LFC is used instead of PFC, TC index 0 is used to request ignoring LFC. Byte 17, bits 6:0 - Reserved, must be zeroed. Byte 17, bit 7 - Ignore flag: When set to 1b, the PF requests to ignore PFC conditions on the TC indexes set to 1b in the TC bitmap. When clear to 0b, the PF requests to release any ignore PFC condition request issued for the TC indexes set to 1b in the TC bitmap.
Reserved	18-19		Reserved, must be zeroed.
Reserved	20-23		
Reserved	24-27		
Reserved	28-31		

7.7.5.2 LLDP/DCBX Admin Commands

Refer to the LLDP Protocol commands described in [Section 7.12.5.2.3](#).



7.7.5.3 DCB Updated Command

When LLDP is handled by the PF, this command is used to notify EMP that a DCB setting has been modified.

When LLDP is handled by EMP, it is used by PF to notify EMP that one of the following parameters has been modified:

- PFCLinkDelayAllowance set by PRTDCB_GENC.PFCLDA
- PCIRTT set by PRTDCB_GENC.PCIRTT

In return, the EMP might modify the mapping of untagged traffic (via PRTDCB_RUP). A command completion is posted once the mapping has been changed.

Table 7-145. DCB Updated Response

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0302	Command opcode.
Datalen	4-5		Must be zeroed.
Return value	6-7		Return Value: 0x0 - if needed, RPB or untagged traffic mapping was modified accordingly.
Cookie High	8-11	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Reserved	16		Reserved, must be zeroed.
Reserved	17-19		
Reserved	20-23		
Reserved	24-27		
Reserved	28-31		

Table 7-146. DCB Updated Command

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0302	Command opcode.
Datalen	4-5		Must be zeroed by the software device driver.
Return value	6-7		Must be zeroed by the software device driver.
Cookie High	8-11	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Reserved	16-19		Reserved, must be zeroed.
Reserved	20-23		
Reserved	24-27		
Reserved	28-31		



7.7.5.4 Set DCB Parameters Command

This command is used to request that the firmware apply the DCB configuration even when LLDP/DCBx is disabled.

Table 7-147. Set DCB Parameters Command (Opcode 0x0303)

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0303	Command opcode.
Datalen	4-5	0	Must be zeroed by the software device driver.
Return Value	6-7		Must be zeroed by the software device driver.
Cookie High	8-11	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Command Flags	16		Bit 0 = Link-up DCB configuration mode for the port: 0b = Firmware does not apply any DCB configuration when a link up event occurs. 1b = Firmware applies the default DCB configuration when a link up event occurs.
Valid Flags	17		Bit 0 = Bit 0 of the command flags is valid. If the <i>Valid Flag</i> bit is 0, no change is made to the link-up DCB configuration mode.
Reserved	18-31	0x0	Reserved.

Table 7-148. Set DCB Parameters Response

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0303	Command opcode.
Datalen	4-5	0	No external buffer.
Return Value	6-7		Return code written by firmware. In the case of a failure, the existing configuration is unmodified.
Cookie High	8-11	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Reserved	16-31		Reserved.

7.7.5.5 LAN Queue Overflow Event

This event is sent from the device to a PF. It does not generate a response.



Table 7-149. LAN Queue Overflow Event

Name	Bytes.Bits	Value	Remarks
Flags	0-1	0	See Section 7.10.5.1.1 for details.
Opcode	2-3	0x1001	Command opcode.
DataLen	4-5	0x00	N/A
Return Value	6-7	0x00	N/A
Cookie High	8-11	Cookie	Opaque value. Copied by the EMP into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the EMP into the completion of this command.
PRTDCB_RUPTQ	16-19		Contains a copy of the PRTDCB_RUPTQ register reporting the absolute index (in the device space) of the reported receive queue.
QTX_CTL	20-23		Contains a copy of the QTX_CTL register of the matched transmit queue pair of the reported receive queue.
Reserved	24-31		Reserved, must be zeroed.

7.8 Transmit Scheduling

7.8.1 Bandwidth Management Hierarchy

7.8.1.1 Hierarchy of Switching Elements

The X710-TM4 contains a number of switching elements that might be arranged in a hierarchical manner. Depending on how the X710-TM4 is configured, the hierarchy might consist of zero, one or two layers. Each switching element contains a single uplink (egress) port and one or more virtual (ingress) ports. The egress port of the switching element might be connected to the physical port or to the virtual ingress port of another switching element. For a detailed description of the switching components, terminology and hierarchy rules supported by the X710-TM4, see [Section 7.4.2.1](#).

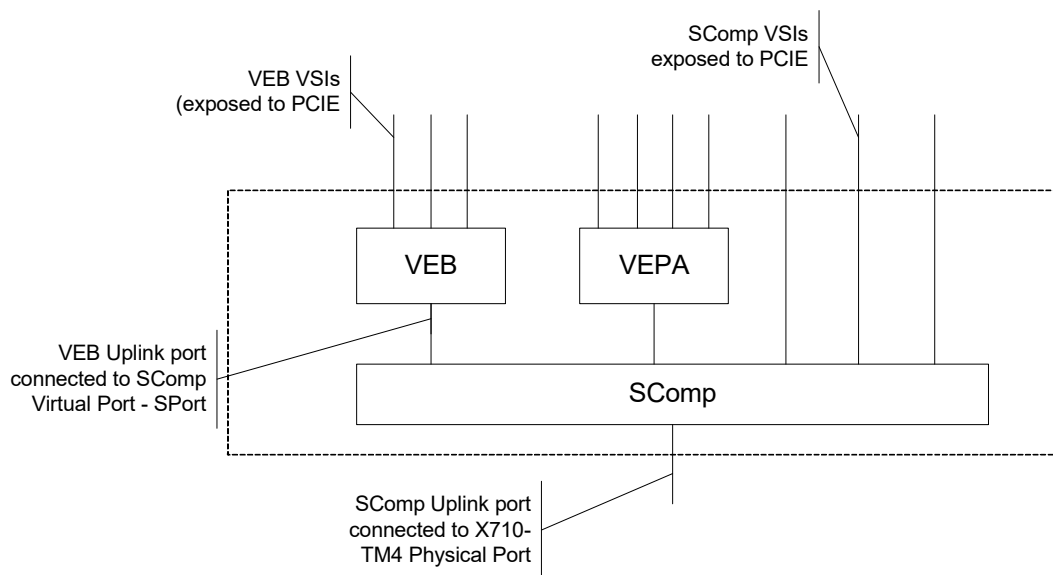


Figure 7-66. Hierarchy of Switching Elements

Figure 7-67 shows an example of a two level switching hierarchy. The S-comp ingress port is connected to the physical port. Several of the S-comp virtual ports (S-channels) are exposed as PCIe functions. Two of the S-comp egress ports are connected to VEB and VEPA switching elements. Both the VEB and VEPA switching elements expose multiple virtual ports as PCIe functions, or VSIs.

A virtual port that is not connected to the egress of a switching element is exposed as a virtual port to the PCIe interface, and is called a Virtual Station Interface (VSI). In this chapter, we use the term virtual port to refer to all ports of a switching component, and the term VSI for virtual ports exposed as PCIe functions. Each VSI is assigned to either the Physical Function (PF) or Virtual Function (VF) and connected to the virtual egress port of the respective PCIe function. A VSI might be owned by a single PCIe function at any one time, but might transition from one PCIe function to another. A single PCIe function might own multiple VSIs.



By default, a VF owns a single VSI. A virtual machine that needs connectivity to multiple virtual ports or switching elements can either use an emulated path through a VMM (using VSIs owned by the PF), or use multiple VFs, one for each VSI. More details on the association of various switching elements and VSIs and PCIe functions can be found in [Section 7.4.2.1](#).

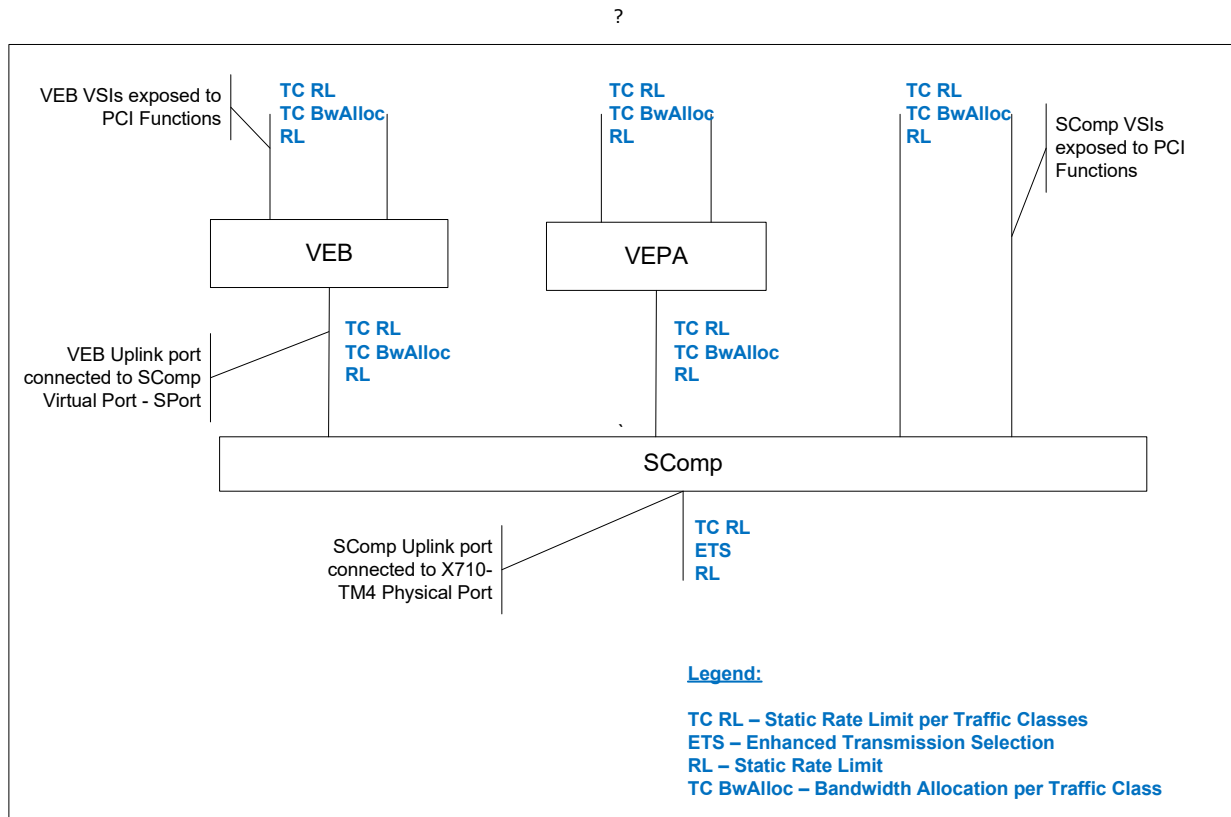


Figure 7-67. X710-TM4 Bandwidth Management Attributes

The X710-TM4 transmit scheduler configuration follows the topology of the internal switching components. [Figure 7-68](#) shows various bandwidth management attributes and their association with the switching components. A subset of bandwidth management controls shown on this diagram is available, depending on the transmit scheduler configuration scheme. See [Section 7.8.2](#) for description of supported transmit scheduler configuration schemes.

- VSI/switching component bandwidth limit. Each VSI and switching component can be configured to limit maximum bandwidth used by that VSI and switching component. Limiting bandwidth of a switching component effectively limits the bandwidth for all VSIs associated with that switching component. See [Section 7.8.1.3](#). Bandwidth limits of VSIs and switching components can be configured in all supported transmit scheduler configuration schemes. See [Section 7.8.2](#).



- ETS configuration of physical port. The X710-TM4 physical ports can be configured using ETS configuration. This enables the specification of the distribution of bandwidth among TCs enabled for the physical port. Each physical port can be configured with an independent ETS. TCs configured to the physical port can be enabled for switching components and VSIs allocated for that port. Allocation, association and bandwidth distribution within each of the TCs must be consistent with physical port ETS configuration and bandwidth distribution within the switching hierarchy. For example: TCs enabled for a VSI must be enabled for respective switching components and physical ports, and the total bandwidth allocated to all VSIs for a particular TC must be equal to the bandwidth allocated for that TC on the switching component. For details on ETS configuration, see [Section 7.8.1.4](#).
- Per traffic type bandwidth allocation. Each VSI and switching component can have relative bandwidth allocated per TC. Such bandwidth allocation is hierarchical, like VSI/switching component bandwidth allocation. But it is done within a particular Traffic Class (TC). For example, each VSI of a switching component can be configured with a bandwidth share with respect to other VSIs of the same switching component for that particular TC. Such bandwidth allocation is mutually exclusive of the VSI bandwidth allocation, and depends on the transmit scheduler configuration scheme. See [Section 7.8.2](#). The bandwidth allocation principles are described in [Section 7.8.1.2](#).
- Per traffic type bandwidth limit. Each VSI that has an ETS enabled can be configured to limit the maximum bandwidth available for one or more TCs configured for that port. For details, see [Section 7.8.1.4](#).

Configuration of the the X710-TM4 scheduler is done by firmware. Software can use the Admin Queue interface to modify the default allocation of bandwidth attributes. See [Section 7.8.3.1](#).

[Table 7-150](#) lists the bandwidth management attributes supported for each switching component and VSI, depending on the transmit scheduler configuration scheme. For a description of both transmit scheduler configuration schemes, see [Section 7.8.2](#).

Table 7-150. X710-TM4 Bandwidth Management Attributes

Configuration Scheme	Switching Element Name	Bandwidth Attribute	Description
ETS-based Scheme	S-comp	Bandwidth limit	This applies to any switching component or VSI directly attached to the physical port. Bandwidth limit for entire physical port regardless type of the traffic (TC).
		ETS	Relative bandwidth allocation between TCs within switching component.
		Traffic type bandwidth limit	Bandwidth limit for individual TC.
	VEB/PA	Bandwidth limit	Bandwidth limit for entire switching component regardless type of traffic (TC).
		Traffic type bandwidth allocation	Relative bandwidth allocation of switching component within each traffic type (TC).
		Traffic type bandwidth limit	Bandwidth limit for switching component per traffic type (TC).
	VSI	Bandwidth limit	Bandwidth limit for entire VSI regardless type of traffic (TC).
		Traffic type bandwidth allocation	Relative bandwidth allocation of VSI within each traffic type (user priority (UP)).
		Traffic type bandwidth limit	Bandwidth limit for VSI per traffic type (TC or UP).



7.8.1.2 Bandwidth Allocation

The X710-TM4 supports configurable allocation of bandwidth to the switching elements and their virtual (ingress) ports. Each switching element gets a certain portion of bandwidth allocated to its uplink port. The total bandwidth available for the switching component is shared across its ingress virtual ports. This effectively allows distribution of the egress switch port bandwidth between ingress ports, or VSIs in case of virtual port exposed to PCIe functions. Each virtual port is configured with its share of the switching element bandwidth.

The default configuration assumes equal bandwidth distribution between ingress ports. This default configuration can be changed using the Admin Queue commands defined in [Section 7.8.4](#). By default, bandwidth distribution between switch virtual ports is relative, and if some of the virtual ports do not use their bandwidth allocation, the remaining bandwidth can be used by other virtual ports relative to their original bandwidth share. For example, if bandwidth allocation of three VSIs is configured to 10%, 30% and 60% respectively; and if VSI_2 (configured to 60%) does not use its bandwidth; then VSI_0 and VSI_1 can share the remaining 60% of the wire bandwidth relative to their original bandwidth allocation (which was 10% for VSI_0 and 30% for VSI_1). The new bandwidth distribution would be 25% for VSI_0 and 75% for VSI_1.

Note: The bandwidth ratio between the two VSIs remains the same (1:3). In relative bandwidth allocation mode, all entities that can be scheduled are assigned a non-zero bandwidth share. See [Figure 7-68](#) for an example of bandwidth distribution.

Bandwidth allocation is not accumulative. If a virtual port does not use its bandwidth share for any reason (such as no work available, bandwidth limit, priority flow control, etc.), it does not accumulate additional bandwidth. Next time it becomes active, it is allowed to consume its allocated bandwidth share, regardless of having stayed idle and not using bandwidth.

The granularity of bandwidth allocation per virtual link is 1% of the bandwidth available for that virtual link. The virtual port propagates its bandwidth allocation to the uplink egress port of the switching element, or to the PCIe function owning this virtual port. This bandwidth allocation is shared by virtual ports/VSIs of the switching element, or by the transmit queues of the respective PCIe function.

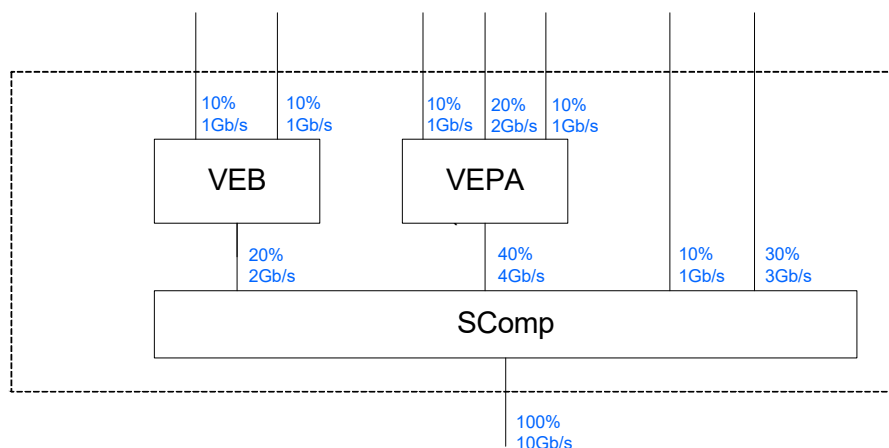


Figure 7-68. Example of Bandwidth Distribution



Figure 7-68 shows an example of bandwidth distribution within the switching hierarchy. The S-comp uplink port is connected to the 10 GbE physical port of the chip, and owns all the bandwidth available on the wire. S-comp has four virtual ports - two VSIs and two S-channels connected to the VEB/VEPA switching components, and the bandwidth allocation for each port is shown on the diagram. Note that bandwidth distribution is relative, and each virtual port gets a percentage of the bandwidth available for the S-comp. For example, if the total amount of bandwidth available for S-comp were decreased to half, bandwidth available for each virtual port would be halved. If some virtual ports do not use all their allocated bandwidth, other virtual ports can use the remaining bandwidth. The VEB switching component is configured to get 20% of the bandwidth available for the S-comp. The bandwidth available for the VEB is in turn equally distributed among the VEB VSIs.

7.8.1.2.1 Arbitration Schemes

The X710-TM4 supports three types of arbitration schemes: Weighted strict priority, weighted round robin and a combination of both. The arbitration scheme is specified by software as a part of the bandwidth distribution definition.

A weighted strict priority scheme allows software to assign different bandwidth share priorities to the entities that can be scheduled. Multiple entities can be configured with weighted strict priority. An entity with higher priority is allowed to fully consume its bandwidth share before a virtual port with lower priority can use a bandwidth share allocated to it. This scheme allows one or more lightly loaded ingress virtual ports with high bandwidth share priority to use bandwidth as soon as data becomes available on those ports, independent of the ports with lower priority. If a virtual port is configured to weighted strict priority, software can specify its bandwidth share to be unlimited by allocating 127 bandwidth share credits to the virtual port. The strict priority arbitration scheme is a weighted strict priority, with unlimited bandwidth allocation credits.

A weighted round robin scheme allows virtual ports to use allocated portions of bandwidth in a round robin fashion. In this configuration, all virtual ports are assumed to have the same bandwidth allocation priority, and ports are interleaving in round robin sequence while using a portion of the bandwidth allocated to the port.

A combined configuration allows software to specify per switching component two sets of ingress virtual ports: one sharing bandwidth using the weighted strict priority scheme, and another sharing bandwidth using the weighted round robin scheme. Virtual ports belonging to the weighted round robin group can use their bandwidth share only if all virtual ports in the weighted strict priority group either fully used their bandwidth share or cannot use their share due to other restrictions (such as no data available for the port, or the port having exceeded its static rate limit).

Software is allowed to modify bandwidth allocation of the virtual port on the fly using the Admin Queue commands described in [Section 7.8.4](#). A new bandwidth allocation might take effect after a few scheduling cycles.

7.8.1.3 Static Rate Limiting

The X710-TM4 enables the configuration of a maximum bandwidth limit for each virtual port of a switching element (both ingress and egress ports), and for the TCs/UPs on ETS/SLA enabled ports. This limit specifies the maximum amount of bandwidth that can be consumed by the virtual port. Limiting the maximum bandwidth of the uplink (egress) port of the switching element effectively limits the total bandwidth that can be used by all virtual (ingress) ports of that switching element. This causes the propagation of the maximal bandwidth limit through the switching hierarchy. Maximum bandwidth allocated for a virtual port should not exceed the maximum bandwidth of the respective uplink port (if it is configured). The sum of the bandwidth limits of all virtual ports of the switching element might exceed the bandwidth limit of the uplink port to allow efficient bandwidth use.



The maximum bandwidth limit can be configured for the ETS traffic class groups and TCs. See [Section 7.8.1.4](#).

Maximum bandwidth limits can be used to share bandwidth between VSIs. This is not the most efficient method, but it is a supported bandwidth distribution scheme. In this case, relative bandwidth allocation should match a maximum bandwidth limit for each port, and the sum of bandwidth limits should add up to the total bandwidth available for the switching element.

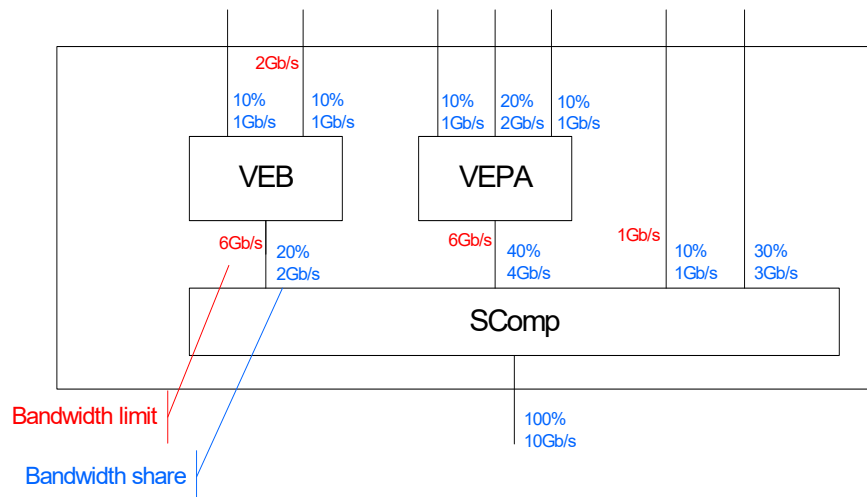


Figure 7-69. Example of Limiting Maximum Bandwidth

Figure 7-70 shows an example of bandwidth distribution between switching elements and their virtual ports in conjunction with limiting the maximum bandwidth available for some of the virtual ports. The bandwidth limit on the uplink port of the VEB limits the total bandwidth available for that switching element (for example, 6 Gb/s for the VEB uplink port on [Figure 7-70](#)). Some virtual ports of the switching element might be bandwidth-limited as well (the right side VSI of the VEB is limited to 2 Gb/s). The bandwidth limit of an individual virtual port should be less than or equal to the bandwidth limit of the switching element or its uplink port. However, for efficient bandwidth use, the sum of bandwidth limits of the virtual ports should exceed a bandwidth limit of the switching element. For example, the sum of bandwidth limits of the S-comp virtual ports is 13 Gb/s, while the total available bandwidth is 10 Gb/s.

The X710-TM4 enables the configurable accumulation of bandwidth limit credits. If a virtual port, with its static rate limit enabled, does not use its allowed bandwidth, that port can accumulate rate limiting credits and use more bandwidth than allowed by the static rate limiter up to the configurable limit for the burst of quanta bytes. The X710-TM4 maintains a single quanta for the entire chip; this is configured via the TSCDQUANTA register.

The X710-TM4 static rate limiters can be configured to the minimal rate limit of 50 Mb/s with a granularity of 50 Mb/s. The maximum rate limit supported by the X710-TM4 is 10 Gb/s.

Static bandwidth limits can be configured to VSIs and switching components in all supported configuration schemes. See [Section 7.8.2](#).



7.8.1.4 Enhanced Transmission Selection (ETS)

In a DCB enabled environment, a single physical port or VSI can be shared by multiple traffic types. The Enhanced Transmission Selection (ETS) standard defines sharing of the virtual or physical link bandwidth by multiple types of traffic. Different types of traffic are segregated into TCs based on User Priorities (UPs). Each TC is mapped to one or more UPs. Mapping UPs to TCs, and sharing the bandwidth of VSI by TCs is defined by the ETS specification. Although ETS bandwidth allocation is defined for the egress ports of the networking devices, it also might apply to the ingress port configuration of adjacent devices sharing the same physical or virtual link.

The X710-TM4 allows ETS to be enabled and independently configured for the uplink ports and VSIs. When enabled, ETS defines the number of TCs allocated per port, the UP to TC mapping, and the distribution of port bandwidth between TCs. Due to the limited amount of scheduling resources, the X710-TM4 limits the number of TCs/UPs that can be installed to an average of two per VSI.

The per traffic type bandwidth allocation of VSIs and physical ports in ETS is independent: each VSI or switching component can be configured with a different number of TCs and with different bandwidth distribution. However, the ETS configuration of the entire internal switching fabric must be consistent. For example, TCs enabled for a VSI must be enabled for respective switching components and physical ports, and the total bandwidth allocated to all VSIs for a particular TC must be equal to the bandwidth allocated for that TC on the switching component.

The X710-TM4 supports a mixed bandwidth allocation for the TCs, whereby some TCs are configured as strict priority TCs and others are configured as ETS TCs. Strict priority TCs have a scheduling priority over ETS TCs, and can consume unlimited bandwidth. Strict priority TCs are intended to be lightly loaded TCs carrying high priority traffic that does not consume much bandwidth. ETS TCs share bandwidth unused by strict priority TCs relative to their allocation of bandwidth share. See [Section 7.8.1.2.1](#) for a description of arbitration schemes.

7.8.1.5 Transmit Queues and Queue Sets Assignment

The X710-TM4 queue set is a set of transmit queues carrying unaccelerated traffic generated by the networking stack.

A VSI might have multiple sets of transmit queues associated with it.

A VSI configured to carry a single type of traffic (like one TC) has one pair of transmit queue sets assigned to it: one stateful and one stateless.

A VSI configured to carry multiple types of traffic has one pair of queue sets allocated for each TC configured for the port.

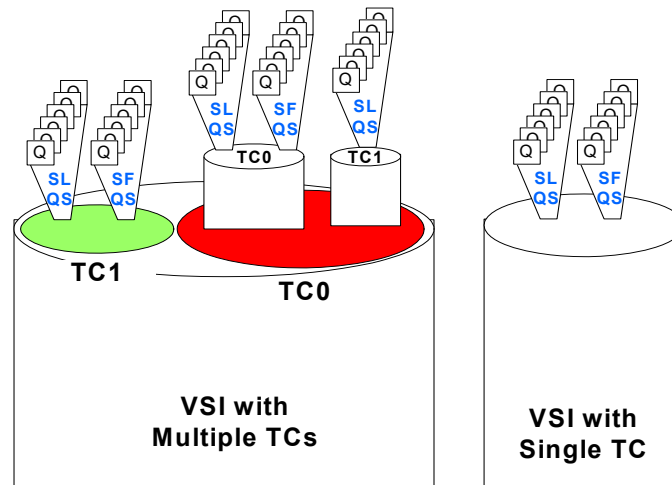


Figure 7-70. Queue Set Assignment Diagram

The allocation of queue sets to VSIs is managed by firmware. This is done either at VSI creation time, or when ETS/SLA configuration of the VSI is enabled or modified by software. See [Section 7.8.4.8](#) and [Section 7.8.4.17](#).

Bandwidth allocation between queue sets in the pair is configured separately, as is bandwidth distribution between VSIs, and ETS bandwidth distribution within VSIs.

All queues assigned to the same queue set belong to the same TC, and have even bandwidth allocation.

Assignment of queues to queue sets is performed by software.

To comply with the function level reset requirement, all queues associated with the same queue set must belong to the same PCIe function.

7.8.1.5.1 Queue Set Reassignment

Certain modifications of scheduler configuration, such as changing number of TCs enabled for VSI ([Section 7.8.4.8](#)), might result in reassignment of active transmit queues from one queue set to another. Transmit queue reassignment involves changing the transmit queue context to carry a new queue set handle.

If both queue sets remain active after a configuration change, then software can perform transmit queue reassignment lazily in the background in parallel with the regular scheduling operation. Transmit queues might get scheduled once via the old queue set before switching to the scheduling using a new queue set.

If an old queue set is deallocated, then software should do the following:

- Upon completion of the configuration change, start reassigning transmit queues to the new queue set.
- Meanwhile, hardware might keep scheduling transmit queues using the old queue set. The re-assigned transmit queue might be scheduled using an old queue set at most once.

- When the transmit queue reassignment is completed, software should notify firmware using the Release Queue Set AdminQ command (Section 7.8.4.9), that it has completed its part of transition and that firmware can proceed with queue set deallocation.

Note: Software should not request release of the queue set, if it still has transmit queues assigned to it.

7.8.2 Transmit Scheduler Configuration Schemes

The X710-TM4 scheduler supports a wide variety of configuration schemes. Programming interfaces described in Section 7.8.4 allow software to configure the ETS-based configuration scheme described in Section 7.8.2.1.

7.8.2.1 ETS-Based Scheduler Configuration Scheme

Figure 7-71 shows a logical diagram of the ETS-based configuration scheme.

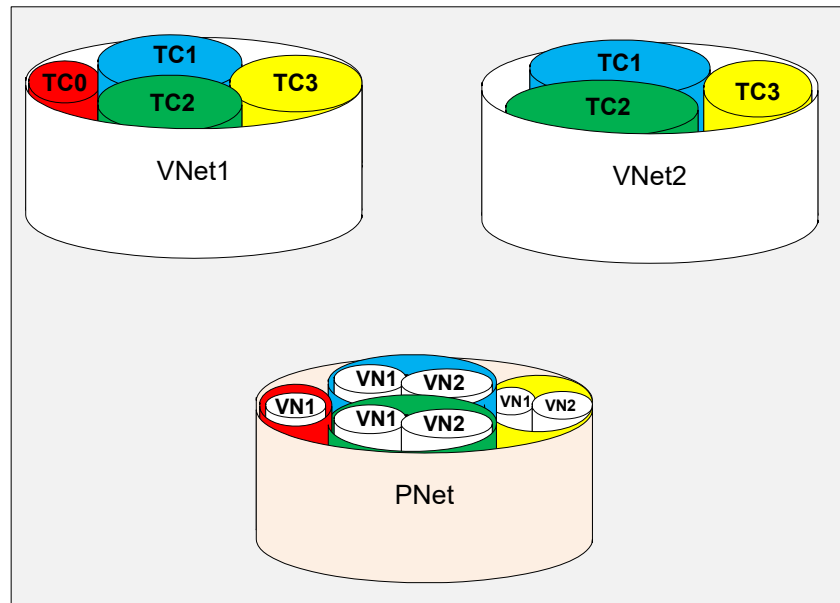


Figure 7-71. ETS-Based Scheduler Configuration Scheme

Figure 7-71 shows two virtual networks. Each virtual network carries multiple types of traffic. Each traffic type can be identified by its TC or UP. Virtual networks are merging to the physical network. The physical network shows a bandwidth distribution between virtual networks and their traffic types in this ETS-based scheduler configuration scheme.



In the ETS-based configuration, bandwidth is first distributed between types of traffic on the wire and then between virtual networks within each traffic type. Note that traffic type bandwidth distribution might be more complex than shown in [Figure 7-71](#), and might include multiple bandwidth distribution layers. These might include bandwidth distribution between TCs, and then between UPs within a TC. Similarly, bandwidth distribution between virtual networks might involve multiple bandwidth distribution levels (such as virtual switches, and their virtual ports).

If a virtual network does not have traffic currently available of a particular traffic type (such as VNet2.TC3), then the physical network attempts to preserve the ratio of bandwidth allocated for that traffic type, and the remaining bandwidth is distributed between other virtual networks having same traffic type available (like VNet1.TC3).

Such per-traffic type bandwidth distribution does not guarantee any bandwidth allocation for the virtual network, but it does allow full control over traffic distribution on the wire per traffic type (such as ETS). This scheme does guarantee virtual network bandwidth allocation per type of traffic (TC or UP).

This configuration is suitable for the DCB-enabled fabric. As long as traffic is available for each traffic type enabled on the virtual network, the bandwidth distribution on the physical wire matches the ETS configuration of the physical port.

The ETS-based configuration scheme allows one to configure an ETS for the switching component or VSI connected to the physical port, and hierarchically distribute per-traffic type bandwidth between all other switching components, (such as VEBs, PAs, and port extenders) and VSIs.

[Figure 7-72](#) shows an example of bandwidth distribution in an ETS-based scheduler configuration scheme.

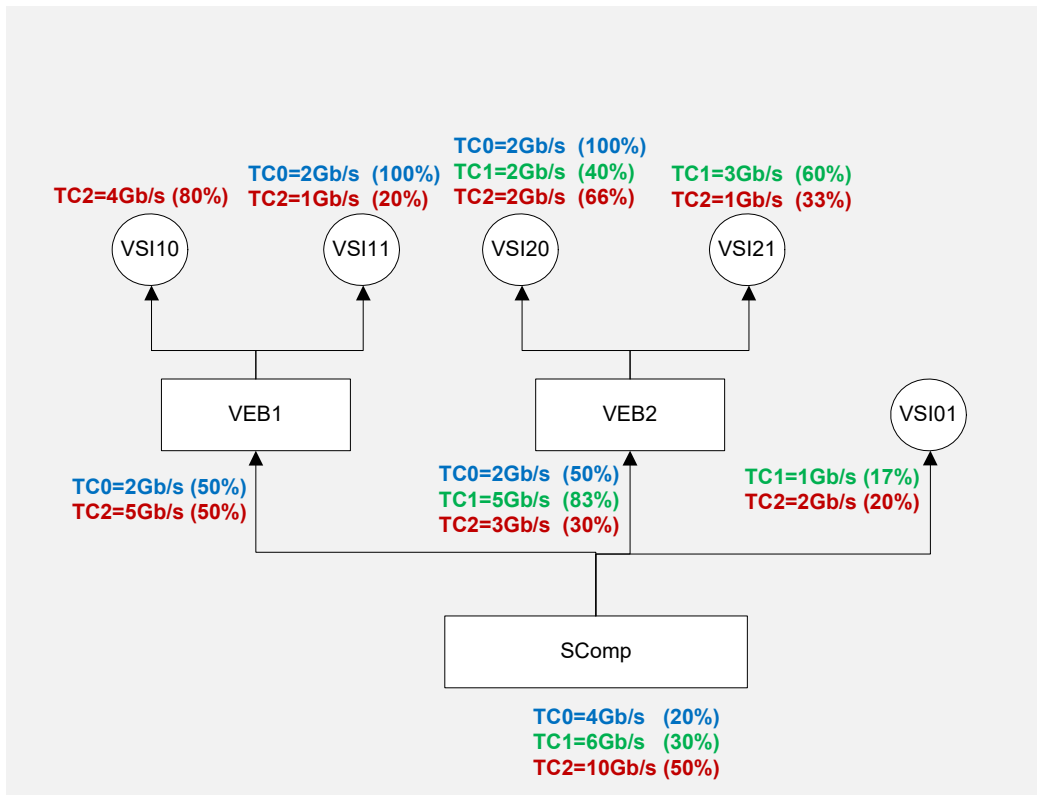


Figure 7-72. ETS-Based Bandwidth Configuration Example

Figure 7-72 shows a single port configuration with S-comp. Two VEB switching components are connected to S-comp (VEB1 and VEB2), along with a VSI (VSI01). Each VEB has two VSIs (VSI10 and VSI11, and VSI20 and VSI21, respectively). Each of the switching components is configured with bandwidth allocation per traffic type. The physical port is configured with ETS, showing one level of bandwidth distribution (per TC). The physical port can be configured with two level ETS, including bandwidth allocation per TC, and bandwidth allocation per UP within the respective TC.

Bandwidth allocation between switching components and VSIs on each level of switching topology is relative within the same TC and is also hierarchical. Figure 7-72 shows an example bandwidth allocation in Gb/s, with relative percentages. Color coding helps to visualize the bandwidth distribution between switching components and VSIs within the TC.

- S-comp level.
- VEB Level (VEB1, VEB2 VSI01).
- Per TC bandwidth is distributed between VEBs and VSI. Bandwidth distribution is relative. Total of 100% of bandwidth per TC.
 - TC0: VEB1=50%, VEB2=50%
 - TC1: VEB2=83%, VSI01=17%
 - TC2: VEB1=50%, VEB2=30%, VSI01=20%



The total effective bandwidth (in Gb/s) allocated for TCs on VEB level is equal to the bandwidth allocated per TC for S-comp. For example, $VEB1.TC0 + VEB2.TC0 = SComp.TC0$, $VEB2.TC1 + VSI01.TC1 = SComp.TC1$, and $VEB1.TC2 + VEB2.TC2 + VSI01.TC2 = SComp.TC2$.

- VSI Level (VSI10, VSI11) and (VSI20, VSI21).
- Per TC bandwidth is distributed between VSIs within the respective VEB. Bandwidth allocation for each TC is relative within VEB.
-
- VEB1 VSIs:
 - TC0: VSI10=100%
 - TC1: 0%
 - TC2: VSI10=80%, VSI11=20%

The total effective bandwidth (in Gb/s) allocated for TCs on VEB1 VSI level (VSI10 and VSI11) is equal to the bandwidth allocated per TC for VEB1. (i.e. $VSI10.TC0 + VSI11.TC0 = VEB1.TC0$, and $VSI11.TC2 = VEB1.TC2$)

VEB2 VSIs:

- TC0: VSI20=100%
- TC1: VSI20=40%, VSI21=60%
- TC2: VSI20=66%, VSI21=33%

The total effective bandwidth (in Gb/s) allocated for TCs on VEB2 VSI level (VSI20 and VSI21) is equal to the bandwidth allocated per TC for VEB2. (i.e. $VSI20.TC0 = VEB2.TC0$, $VSI20.TC1 + VSI21.TC1 = VEB2.TC1$, and $VSI20.TC2 + VSI21.TC2 = VEB2.TC2$).

Bandwidth allocation must be consistent. Bandwidth allocated to the TC on the S-comp level must be equal to the total bandwidth allocated for the same TC on the level of VEBs and VSIs connected directly to the S-comp (in this case, VEB1, VEB2 and VSI01). Bandwidth allocated to the TC on VEB (VEB1) level must be equal to the total bandwidth allocated to that TC on VSI level (VSI10 and VSI11).

The ETS-based scheduler configuration scheme does not support bandwidth allocation per switching component or VSI. It supports only bandwidth allocation for switching component or VSI within particular TC.

The ETS-based scheduler configuration scheme allows software to instantiate bandwidth limits for each of the switching components and VSIs. The bandwidth limit can be programmed to restrict total bandwidth available for switching component and VSI, or limit bandwidth available for the switching component or VSI for the particular type of traffic, identified by UP or TC.

7.8.3 Scheduling

The X710-TM4 scheduler operates with queue sets. Every cycle scheduler selects a next queue set to be served. The amount of traffic that can be generated on the wire by transmit queues associated with the selected queue set is limited to the configurable value - Quanta. Selection of the queue set is based on the bandwidth share, bandwidth limit and ETS configuration of the VSI and uplink ports of the internal switching components. The scheduler fairly distributes available bandwidth among queue sets that have transmit queues with work available, staying within limits of the bandwidth allocation dictated by bandwidth limit, bandwidth share and ETS constraints. Bandwidth distribution between queues within the same queue set is intended to be even.

The scheduler limits amount of data that can be transmitted by selected queue set to the configurable value - Quanta. Single quanta might allow transmission of multiple Ethernet frames. Quanta constrains the burstiness of the traffic generated by the X710-TM4. Independent of the rate limit implied on the



queue set, the X710-TM4 transmits quanta worth of data with the wire speed of the associated physical port. Scheduler configuration is derived from the configuration of internal switching elements, and in most cases follows the hierarchy of the switching elements.

The X710-TM4 does not schedule individual packets. All packets belonging to the same transmit queue will follow the same route within the internal switching hierarchy, and are subject to the same set of bandwidth constraints.

When the X710-TM4 transmits packets from one of the transmit queues of a selected queue set, it effectively uses the bandwidth of all switching components in the internal switching hierarchy leading from the virtual port that queue set is associated with to the physical port of the chip. Bandwidth constraints applied to the switching component on the route from the queue set to the physical port of the chip automatically apply to the queue set. This means that a particular queue set can be scheduled only if there is enough bandwidth available in all switching components on the transmission route.

The scheduler is aware of the PFC, and stops scheduling queue sets associated with paused TC to avoid head of line blocking in the pipeline.

7.8.3.1 Scheduler Configuration Process

The scheduler implements one set of configuration tables shared by all PCIe functions. To synchronize access to the shared configuration table and keep configuration consistent, direct access to the internal scheduler configuration is restricted to firmware.

The scheduler does not expose its internal configuration tables to the standard deployment software. A set of registers allowing a direct access to the internal scheduler structures is exposed to firmware to allow programming of scheduler configuration tables. Those registers are exposed in debug mode to bring up the other privileged software. A default configuration of the scheduling tables is done by firmware, and is based on the internal switch and scheduling configuration profiles kept in NVRAM. Software is allowed to modify the default configuration by proxying its requests through firmware using Admin Queue commands. Software should use an Admin Queue interface to communicate its scheduler configuration requests with firmware. Firmware is responsible for constraining access to the scheduling configuration tables based on the software privilege level.

The X710-TM4 maintains a single quanta for the entire chip, configured via the TSCDQUANTA register. Quanta accounts for the payload and headers generated by software for the LAN traffic and payload. Quanta does not account for the L2 Tags and Ethernet CRC padding inserted by hardware on the fly.

The scheduler is intended to be configured by physical function driver using the Admin Queue commands described in [Section 7.8.4](#). A physical function driver is considered a trusted software, and firmware should accept requested configuration changes. Most of the switching components are owned either by a single PF, or by one of the VFs associated with the same PFs. If a switching component is shared among PFs (such as S-comp in a Multi-function Per Port (MFP) environment), this component is either configured by an EVB agent running in firmware, or by a service running in the PF. In the latter case, firmware relies on software to coordinate the scheduler configuration among different PFs.

In the standard deployment environment, most of the scheduler configuration is fully controlled by the PF driver. The VF driver or configuration software running in VFs is granted control over one of the scheduling configuration parameters. For example, for ETS/SLA configuration of the virtual port, the scheduler configuration commands should be a proxy via PF driver.

The scheduler supports up to 768 pairs of queue sets, regardless of the number of queue sets populated in each pair.



7.8.4 Admin Queue Commands

The internal structure of the scheduler configuration tables is not exposed to software. Firmware is responsible for the scheduler configuration and provides software with an Admin Queue interface allowing the alteration of scheduler configuration. In some configurations, privileged software is allowed to perform direct programming of the scheduler configuration tables by bypassing firmware. Firmware or privileged software should maintain a mapping table of the scheduling resources assigned to each PCIe function.

This section defines Admin Queue commands that should be used by software to program scheduler configuration attributes.

A standard X710-TM4 scheduler configuration is based on the configuration of the internal switch and its components, including instantiated switching components, their VSIs, S-channels, and connectivity between internal switching components. Scheduler configuration adds bandwidth management attributes to the configured switching components, including allocation of bandwidth for the switching components and their VSIs, the enablement and configuration of ETS, or bandwidth allocation per TC, and instantiation of rate limiters.

The Admin Queue commands described in this section are intended to be used by the PF driver only. The VF driver is not allowed to directly participate in scheduler configuration. The PF driver is considered to be a trusted software component within this PF. Firmware will validate that AdminQ commands impacting configuration of the components are owned by the PF driver issuing those commands. This will be done using an association of the Admin Queue with particular PF and information kept in the switch configuration tables. When the PF performs configuration on behalf of the VF, the corresponding VF must be provided within the Admin Queue command.

[Table 7-151](#) provides a list of Admin Queue commands allowing the configuration of internal scheduler structures. The creation of switching components, including VSIs, is described in [Section 7.4.9.4.2](#).

Firmware must avoid partial command execution. All command validation and resource availability checks must be done prior to updating scheduler configuration tables. Aborting commands with a partially updated scheduler configuration table might lead to unpredictable hardware behavior.

Table 7-151. Scheduler Configuration Admin Queue Commands

Command	Opcode	Brief description	Detailed Description
Configure VSI Bandwidth Limit	0x0400	This command allows software to configure bandwidth limit to the specified VSI of the specified switch component. This is a global bandwidth limit that applies to all TCs enabled for VSI.	Section 7.8.4.6
Configure VSI Bandwidth Limit per Traffic Type	0x0406	This command allows software to configure bandwidth limits assigned to TCs of the specified VSI exposed to PCIe interface	Section 7.8.4.7
Configure VSI Bandwidth Allocation per Traffic Type	0x0407	This command allows software to specify which Traffic Types are enabled for VSI, and configure relative bandwidth allocation of VSI within each traffic type (TC). This command is valid for ETS-based configuration only.	Section 7.8.4.6
Query VSI Bandwidth Configuration	0x0408	This command allows software to retrieve current configuration of the specified VSI of the specified switching component. This configuration should include bandwidth limit and bandwidth allocation.	Section 7.8.4.15
Query VSI Bandwidth Configuration per Traffic Type	0x040A	This command allows software to retrieve current bandwidth configuration of VSI within each Traffic Type (TC). This command is valid for ETS-based configuration only.	Section 7.8.4.16



Table 7-151. Scheduler Configuration Admin Queue Commands

Command	Opcode	Brief description	Detailed Description
Configure switching component Bandwidth Limit	0x0410	This command allows software to enable, disable or modify the bandwidth limit of the egress port of a specified switching component. This is a global bandwidth limit that applies to all TCs enabled for a switching component.	Section 7.8.4.9
Enable physical port ETS	0x0413	This command allows software to enable ETS configuration of the egress port of the specified switching component or VSI directly connected to the physical port. It carries full specification of ETS configuration for the port, including number of TCs. UP to TC mapping, TC bandwidth allocation, TC arbitration scheme, and bandwidth allocation. This command is valid only for switching components directly connected to the physical port. This command is valid in ETS-based configuration only.	Section 7.8.4.10
Modify Physical Port ETS	0x0414	This command allows only to modify ETS configuration of the egress port of the specified switching component or VSI directly connected to the physical port. It carries full specification of ETS configuration for the port, including number of TCs. UP to TC mapping, TC bandwidth allocation, TC arbitration scheme, and bandwidth allocation. This command is valid only for switching components directly connected to the physical port. This command is valid in ETS-Based configuration only.	Section 7.8.4.10
Disable Physical Port ETS	0x0415	This command allows only to disable ETS configuration of the egress port of the specified switching component or VSI directly connected to the physical port. It carries full specification of ETS configuration for the port, including number of TCs. UP to TC mapping, TC bandwidth allocation, TC arbitration scheme, and bandwidth allocation. This command is valid only for switching components directly connected to the physical Port. This command is valid in ETS-based configuration only.	Section 7.8.4.10
Configure switching component Bandwidth Limit per Traffic Type	0x0416	This command allows software to enable, disable or modify bandwidth limits assigned to TCs of the egress port of the specified switching component. This command is valid in ETS-based configuration only.	Section 7.8.4.13
Configure switching component Bandwidth Allocation per Traffic Type	0x0417	This command allows software to specify traffic types enabled for switching components and configure relative bandwidth allocation of switching component within each traffic type (TC). This command is valid for ETS-based configuration only.	Section 7.8.4.13
Suspend Port's TX Traffic	0x041B	This command allows the PF to suspend a port's transmit traffic. The command is completed after all queue sets belong to the port are suspended and the Tx pipe of the port is drained. This allows software to modify a port's configuration like the DCB setting. This command is valid only under SFP mode.	Section 7.8.4.11
Resume PF Traffic	0x041C	This command is used to resume suspended queue sets. It will resume all queue sets that belong to the PF.	Section 7.8.4.12
Query switching component Bandwidth Configuration	0x0418	This command allows software to retrieve the current configuration of the switching component.	Section 7.8.4.17
Query Physical Port ETS Configuration	0x0419	This command allows software to retrieve current ETS configuration of the switching component. This should include the number of valid TCs, bandwidth allocation and bandwidth limit for TCs. This command is valid for switching components directly connected to the physical port only. This command is valid in ETS-based configuration only.	Section 7.8.4.18
Query switching component Bandwidth Configuration per Traffic Type	0x041A	This command allows software to retrieve the current bandwidth configuration of the switching component within each Traffic Type (TC). This command is valid for ETS-based configuration only. This command is valid only under SFP mode.	Section 7.8.4.19

**Table 7-151. Scheduler Configuration Admin Queue Commands**

Command	Opcode	Brief description	Detailed Description
Configure Tx Port Settings	0x041E	This command is used by software to configure which port works in Multi Qset mode and which in legacy Uni-Qset mode	Section 7.8.4.20
Add VSI	0x0210	This is a switch configuration command. Execution of this command affects the scheduler configuration. This command allocates VSI and returns a VSI SEID that should be used to modify the VSI bandwidth configuration.	Section 7.8.4.2
Add VEB and Add Port Virtualizer	0x0210	This is a switch configuration command. Execution of this command affects scheduler configuration. This command allocates the switching component, and returns a SEID that should be used to modify the switching component bandwidth configuration.	
Delete Element	0x0210	This is a switch configuration command. Execution of this command affects the scheduler configuration. This command deallocates switching elements including VSIs. Deallocation of the switching element results in deallocation of the scheduling resources associated with this switching element, and cleanup of scheduler configuration tables.	

7.8.4.1 Scheduler Initialization Flow

Main scheduler configuration and initialization flows are initiated by software, and performed by firmware using AdminQ interface. The following sections define various AdminQ commands that can be used by software to configure the transmit scheduler.

Prior to the initialization of the Admin Queue Command interface, firmware can use the information available in NVRAM to perform default configuration of the scheduler. Similar to the standard configuration, default scheduler configuration follows a default configuration of the switching components.

7.8.4.2 Allocation of Switching Elements

A default configuration of the scheduler tables is performed as a part of the internal switch configuration. Each time software adds a new switching component ([Section 7.4.9.5.6.1](#)) or VSI to the internal switch, the configuration tables of the scheduler are updated respectively by firmware. A new added switching component or VSI is configured with a default bandwidth allocation and bandwidth limit disabled. The allocated switching component or VSI is created with a single TC enabled (TC0). Software can enable TCs either by using the bit mask in the Add VSI or Add VEB/PA commands, or by using the AQ command described in [Section 7.8.4.7](#) and [Section 7.8.4.14](#). Along with default bandwidth, distribution firmware allocates a pair of queue sets for each configured TC of the newly allocated VSI.

Upon completion of a switching element allocation request, software is provided with a handle per allocated queue set. This queue set handle should be used to associate the queue with a queue set.

A default bandwidth allocation provides a new instantiated switching component and VSI with a minimal bandwidth share with respect to other switching components or ports sharing same virtual link. Software can modify the default bandwidth allocations using the Admin Queue command described in [Section 7.8.4.8](#) and [Section 7.8.4.14](#).

The initial configuration of the scheduler tables is done as part of the initial configuration of the internal switch, and is based on the chip profiles. No software involvement is required to perform initial scheduler configuration. As soon as the firmware completes its initial configuration, the scheduler is enabled and ready to schedule transmit work.



Software is allowed to change internal switch configuration or modify scheduler configuration by adding/removing the internal switching components and/or VSIs, or by changing bandwidth distribution, bandwidth limits and ETS configuration of Physical Port, switching components or VSIs. Some configuration changes, causing modification of the internal switching structure, might require firmware to partially or fully suspend scheduler operation until it completes a local or global modification of the internal scheduler tables. Certain scheduler configuration changes, such as bandwidth redistribution, or modification of the bandwidth limit, can be done without suspending normal scheduler operation.

Firmware is responsible for tracking allocated resources and their configuration, and for managing the configuration of internal scheduler tables. If software performs incremental modification to the internal switching structure, it should rely on firmware to maintain consistency of the internal scheduler tables, and fit requested modifications to the existing configuration. Each allocated switching element, including instances of internal switch or VSI, is assigned a unique-within-the-chip identification number - SEID. For the description see [Chapter 7.4.2.1](#). The SEID should be used by software when referencing the switching component during scheduler configuration.

7.8.4.2.1 Queue Set Assignment

As part of the default scheduler configuration, each instantiated VSI is supplied with a pair of queue sets per configured TC - one for stateless and one for the stateful queues. Firmware manages shared pool of queue sets. The allocation and deallocation of queue sets is hidden from software, and is done based on software resource allocation/deallocation requests. Handles of allocated queue sets are returned to software as a part of completion of the Admin Queue command and should be used by software to associate transmit queues and queue sets. Firmware allocates a single queue set handle to the LAN queue sets pair.

Software can change a default configuration of the VSI. (It can change the number of configured TCs). A change in VSI configuration results in changing the number of TCs allocated for the port, causing firmware to reallocate queue sets assigned to the VSI to match number of TCs. When the port is configured with Multi_Qset, each enabled TC instance of the VSI is associated with a group of eight queue sets.

Allocated queue set handles are provided at the completion of each Admin Queue command. Firmware always returns 8 queue set handles. The order of handles returned matches the order of TCs enabled for VSI (from 0 to 7). Invalid queue set handles carry a value of 0xFFFF. Queue set handles are returned for all Admin Queue commands that can result in allocation or change of allocation of queue sets: (Create VSI, Configure VSI Bandwidth Allocation per Traffic Type, Configure VSI Bandwidth Limit per Traffic Type). For more details see [Section 7.8.4.7](#) and [Section 7.8.4.8](#). Software can also use Admin Queue commands to query queue sets allocated for a particular VSI, see [Section 7.8.4.16](#).

When the port is configured with Multi_Qset, each one of the returned Queue set assignment is completely hidden from software. Software should use a queue set handle when referring to the particular queue set.

Assignment of queues to queue sets is not described here, and is outside of the scope of the scheduler configuration definition. If software changes configuration of the VSI and this change results in reallocation of queue sets, as long as TC remains assigned to the VSI, all queues associated with the queue set identified by the VSI and TC will remain associated with the queue set. If software modifies the VSI configuration and eliminates the TC which has active associated queues, software must resolve contention and reassign queues.



7.8.4.3 Release of the Switching Elements

Software can use the AQ command Delete Element and the Configure Bandwidth Allocation/Limit of the VSI/switching component per traffic type to remove switching components and VSIs, or to change their configuration per traffic type. Removal of the switching component or VSI is allowed only if the respective uplink switching element has been previously removed, or if a traffic type configuration has been previously adjusted to the uplink switching element. For example, if software changes VEB configuration per traffic type and reduce number of traffic types enabled for a VEB, it must first respectively adjust traffic types for all VSIs allocated for that VEB. Software might request to remove VEB, but only after it has removed all VSIs allocated for that VEB.

Software might request to remove a VSI only if the queue set associated with that VSI has all transmit queues removed, as described in [Section 7.8.4.3.1](#).

7.8.4.3.1 Queue Set Release

Certain modifications to the scheduler configuration, such as a change in the number of TCs enabled for the VSI, or a change to TC mapping or VSI deallocation, might lead to release of the queue set previously allocated for a VSI.

Prior to performing any operation that might require release of the queue set, software must remove from the queue set all queues that were previously associated with the queue set. For the LAN queues, this operation requires that the queue be disabled. In any case, software must either suspend or disable all queues that were previously associated with the queue set. Completion of queue disable operation must be confirmed by hardware.

7.8.4.4 When the port is configured with Multi_Qset mode enabled, each VSI instance includes also eight children nodes and eight **Common Processing and Error Handling**

The generic structure of the Admin Queue command is defined in [Section 7.10.5](#). All Transmit Scheduler Admin Queue commands described in the following sections are derived from the generic Admin Queue commands, and use a generic Admin Queue command structure as a baseline.

Sections describing Transmit Scheduler Admin Queue commands relate to the Admin Queue command fields specific to each particular command. For the description of the common Admin Queue command fields, see [Section 7.10.5](#).

The transmit scheduler uses various types of Admin Queue commands. Some of them are direct Admin Queue commands, wherein all command information is provided within the body of command, and some are indirect commands, wherein additional data is provided within the buffer referred by the Admin Queue command. Some commands report completion within the Admin Queue command body, and others carry completion information within a buffer provided by the original command. Each Transmit Scheduler Admin Queue command indicates its type in the command description. For the detailed description of all Admin Queue command types and their differences, see [Section 7.10.5](#).

Transmit Scheduler Admin Queue commands use the generic error reporting structure described in [Section 7.10.7](#). [Table 7-152](#) lists errors specific to Transmit Scheduler Admin Queue commands and reported in Admin Queue completions, providing their cause and a reported error code.



Table 7-152. Transmit Scheduler Admin Queue Completion Errors

Error Name	Error Code	Description
Invalid handle	ENOENT	Upon allocation of transmit scheduler resources, software is provided with various handles, such as TC Handle, UP Handle, QS Handle. Those handles must be used by software for future reference to allocated resources. This error indicates that the handle provided by software is not valid.
Invalid SEID	ENOENT	SEID provided within Admin Queue command is not valid.
Parameter out of range	ERANGE	Provided argument does not match its definition, (for example, not within allowed range).
Resource allocation failure	ENOSPC	Failed to allocate Transmit Scheduler resource.
Operation not permitted	EPERM	Depending on the scheduler configuration scheme, certain Admin Queue commands should not be used by software. (For example, in ETS-based configuration software should not attempt configure bandwidth allocation to a VSI or switching component). This error indicates that software has attempted an Admin Queue command that is not valid.
Resource is busy	EBUSY	Software attempted to release-a queue set that had associated queues.

7.8.4.5 Function Level Reset

Transmit Scheduler configuration tables are updated by firmware as a part of the function level reset processing by the X710-TM4. All scheduling resources allocated for a particular function are released during this process, including VSIs and queue sets. Respective resources can be allocated to the different PCI functions later on.

The scheduler operation is not suspended during function level reset, but its performance and bandwidth distribution might be intermittently affected. Upon completion of function level reset processing, the transmit scheduler will complete its normal operation.

7.8.4.6 Configure VSI Bandwidth Limit

This command allows software to configure a bandwidth limit for the specified VSI of the switch component that is exposed to the PCIe interface. To configure a bandwidth limit of the egress port, software should use the commands described in [Section 7.8.4.9](#). The X710-TM4 provides hierarchical bandwidth limits that apply to VSI and all associated TCs and UPs. See [Section 7.8.1.3](#).

Software cannot have both bandwidth limit and bandwidth limit per traffic type enabled for the same VSI. A request to enable bandwidth limit for VSI that has bandwidth limit per traffic type enabled should fail and report an EPERM error.

Under MFP, if the VSI is connected directly to a port extender such as an Embedded Management Processor (EMP), firmware ignores the bandwidth parameters from the Admin Queue command and instead retrieves the parameters of TC enable and bandwidth configuration per PF from the MFP Switch configuration in Alternate RAM data, along with the UP to TC mapping.

In that case, if the PF uses more than one TC (for example, if the iSCSI PF, which must be opened for Storage TC and also LAN TC,) the following conditions are present:

- The PF is allowed to enable more than one TC according to the enabled UPs and their UP to TC mapping.



- Bandwidth relative allocation defined for this PF in the Alternate RAM data will be configured for each one of the used TCs.
- Maximum bandwidth for the PF will be declared as a shared rate limiter.

This is a direct Admin Queue command with completion reported within the completion descriptor structure. [Table 7-153](#) lists the command format and command-specific fields.

Table 7-153. Configure VSI Bandwidth Limit Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0400	Command opcode
Datalen	4-5	0	Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
VSI SEID	16-17		Unique identifier of the VSI.
Reserved1	18-19	0	Reserved. Must be set to 0.
Bandwidth Limit Credits	20-21		Bandwidth limit in Mb/s. Supported range is 50 Mb/s - 10 Gb/s, in increments of 50 Mb/s 0 indicates that bandwidth limit is disabled One credit corresponds to bandwidth limit of 50 Mb/s
Bandwidth Limit Max	24.0-24.2		Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of Quanta-worth credits: 0,1, 2, and 4 Quanta worth credits. This field is valid, only if bandwidth limit is enabled
Reserved2	24.3-31	0	Reserved. Must be set to 0.

7.8.4.7 Configure VSI Bandwidth Limit per Traffic Type

This command allows software to specify TCs enabled for a VSI, and to configure a bandwidth limit of TCs enabled for the specified VSI of the switching component.

In an ETS-based configuration scheme, software can configure either VSI bandwidth limit or VSI bandwidth limit per traffic type.

Software should keep track of already configured bandwidth limits and provide complete bandwidth limit configuration, and not just relying on incremental modifications.

Software is allowed to change bandwidth limits per traffic type, for a TC enabled VSI that has VSI bandwidth allocation configured. A new TC enabled for VSI will have a default bandwidth allocated. Software should use the Configure VSI Bandwidth Allocation per VSI AQ Command to modify the default bandwidth allocation.

Software should not request change of TC configuration of VSI leading to release or remapping of TCs, prior to disassociating queues with affected queue sets. If software requests to release queue sets that still have associated queues, the operation fails with EBUSY error.



The only exception for this rule is the case where only one TC needs to be remapped. In this case, software is allowed to use this command when this queue set is suspended and the FVL transmit pipeline is drained from packets that belong to this queue set. After the TC remapping takes place, software might resume its transmit activity using the Resume PF Traffic command described in Section 7.8.4.12. EMP firmware is required to keep the remapped TC’s queue set handle for its new location. the X710-TM4 supports this transmit pipe draining at port-level granularity. This is used by DCBX flows Section 7.8.5.6.1.

Under MFP, if the VSI is connected directly to the port extender, the EMP firmware ignores the TC enable and bandwidth parameters of the AQC, and retrieves the parameters of TC enable and bandwidth configuration per PF from the MFP switch configuration in its Alternate RAM data and UP to TC mapping.

In that case, if the PF uses more than one TC (such as iSCSI PF which must be opened for storage TC and LAN TC as well), the following conditions will be present:

- The PF is allowed to enable more than one TC. According to enabled UPs and UP to TC mapping.
- Bandwidth relative allocation defined for this PF in the Alternate Ram will be configured for each one of the used TC's.
- Maximum bandwidth for the PF will be declared as shared rate limiter.

If TC re-configuration is needed while the VSI is suspended, newly generated TC nodes of this VSI will be suspended too.

Software cannot have both Bandwidth Limit and Bandwidth Limit per Traffic Type enabled for the same VSI. Request to enable Bandwidth Limit per Traffic Type for VSI that has Bandwidth Limit enabled should fail and EPERM error reported in completion.

This is an indirect Admin Queue command. Software should provide a valid buffer carrying additional command attributes as listed in Table 7-155.

Table 7-154 lists command format, and defines fields specific to the command.

Table 7-154. Configure VSI Bandwidth Limits per Traffic Type Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0406	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
VSI SEID	16-17		Unique identifier of the VSI.
Reserved1	18-23	0	Reserved. Must be set to 0.
Data Address high	24-27		Address of buffer.
Data Address low	28-31		

Table 7-155 lists the format of the command buffer and additional command attributes.



Table 7-155. Configure VSI Bandwidth Limit per Traffic Type Command Buffer

Category	Byte/Bit	Field	Description
TC Valid	0.0-0.7	TC0-TC7 Valid	Valid bit per TC. Should list all TCs enabled for VSI. Must be consistent with Physical Port and uplink switching component.
Reserved1	1-15	0	Reserved. Must be set to 0.
TC Bandwidth Limit Credits	16-17	TC0 Bw Limit Credits	This field specifies a bandwidth limit per TC in Mb/s. Supported bandwidth limit range is 50 Mb/s. Bandwidth limit of 0Mb/s indicates that bandwidth limit is disabled. Bandwidth limit should be provided for the valid TC handles only. One credit corresponds to bandwidth limit of 50Mb/s
	18-19	TC1 Bw Limit Credits	
	20-21	TC2 Bw Limit Credits	
	22-23	TC3 Bw Limit Credits	
	24-25	TC4 Bw Limit Credits	
	26-27	TC5 Bw Limit Credits	
	28-29	TC6 Bw Limit Credits	
	30-31	TC7 Bw Limit Credits	
TC Max Bandwidth Limit	32.0-32.2	TC0 Max Bw Limit	Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of quanta-worth credits: 0,1, 2, 4, 8,16 32 and 64 quanta worth credits.
	32.3	Reserved	
	32.4-32.6	TC1 Max Bw Limit	
	32.7	Reserved	
	33.0-33.2	TC2 Max Bw Limit	
	33.3	Reserved	
	33.4-33.6	TC3 Max Bw Limit	
	33.7	Reserved	
	34.0-34.2	TC4 Max Bw Limit	
	34.3	Reserved	
	34.4-34.6	TC5 Max Bw Limit	
	34.7	Reserved	
	35.0-35.2	TC6 Max Bw Limit	
	35.3	Reserved	
35.4-35.6	TC7 Max Bw Limit		
35.7	Reserved		
Reserved2	36-63	0	Reserved. Must be set to 0.

Table 7-156 lists the format of completion returned in response buffer.



Table 7-156. Configure VSI Bandwidth Limit per Traffic Type Completion Buffer

Category	Byte/Bit	Field	Description
QueueSet Handles	16-17	QS0 Handle	Handle per queue set. Completion will always carry eight queue set handles. Only TCs that were marked as a valid TCs in the command will have a valid queue set handles returned in completion. Invalid handle is marked by 0xFFFF. Order of queue set handles matches order of TCs. Queue set handles must be used by software to associated transmit queue with queue set.
	18-19	QS1 Handle	
	20-21	QS2 Handle	
	22-23	QS3 Handle	
	24-25	QS4 Handle	
	26-27	QS5 Handle	
	28-29	QS6 Handle	
	30-31	QS7 Handle	

7.8.4.8 Configure VSI Bandwidth Allocation per Traffic Type

This command allows software to specify TCs enabled for VSI, and configure relative bandwidth allocation of VSIs within each TC.

This command can be used for VSI configured in ETS-Based mode only. Otherwise the command would be ignored, and error reported.

To configure VSI bandwidth allocation per Traffic Type, software should provide complete relative bandwidth allocation for all TCs enabled for VSI.

This is an indirect Admin Queue command. Software should provide a buffer that would be used both to provide additional command attributes and for the command completion.

Software is allowed to change bandwidth allocation per traffic type and a TC enabled for VSI that has a VSI bandwidth limit enabled. It is responsibility of firmware to adjust hardware configuration to reflect this change.

Software should not request change of TC configuration of VSI leading to release or remapping of TCs, prior to disassociating queues with affected queue sets. Operation will fail with EBUSY error, if software requested to release queue sets that still have associated queues.

The only exception for this rule is when remapping of only one TC is required. In this case, software is allowed to use this command when this queue set is suspended, and when the FVL transmit pipeline is drained from packets belongs to this queue set. After the TC remapping takes place, software might resume its transmit activity using the commands listed in [Section 7.8.4.14](#). EMP firmware is required to keep the remapped TCs queue set handle for its new location. FVL supports transmit pipe draining at port-level granularity. This is used by DCBX flows in [Section 7.8.5.6.1](#).

Under MFP, if the VSI is connected directly to the port extender, EMP firmware ignores TC Enable and bandwidth parameters of the AQC and will retrieve the parameters of TC enable and bandwidth configuration per PF from MFP Switch configuration in Alternate RAM data and UP to TC mapping.

In that case, if the PF uses more than one TC (such as iSCSI PF that must be opened for the Storage TC and LAN TC as well).

- The PF is allowed to enable more than one TC, according to enabled UPs and UP to TC mapping.
- Bandwidth relative allocation defined for this PF in the Alternate Ram will be configured for each one of the used TCs.



- Maximum bandwidth for the PF will be declared as a shared rate limiter.

If the TC re-configuration is needed while the VSI is suspended, TC nodes for this VSI will be generated in a suspended state.

Table 7-157 lists the command format and the command-specific fields.

Table 7-157. Configure VSI Bandwidth Allocation per Traffic Type Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0407	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
VSI SEID	16-17		Unique identifier of the VSI
Reserved1	18-23	0	Reserved. Must be set to 0.
Data Address high	24-27		Address of buffer.
Data Address low	28-31		This buffer is used both to convey configuration to firmware, and provide software with a list of TC handles.

Table 7-158 lists the format of the command buffer and its command attributes.

Table 7-158. Configure VSI Bandwidth Allocation per Traffic Type Command Buffer

Category	Byte/Bit	Field	Description
TC Valid	0.0-0.7	TC0-TC7 Valid	Valid bit per TC. Should list all TCs enabled for this VSI. Must be consistent with physical port and uplink switching component. At least one TC must be valid.
Reserved2	1-3	0	Reserved. Must be set to 0.
TC Bandwidth Share Credits	4	TC0 Bandwidth Credits	Relative VSI credits within the same TC with respect to other VSIs or the switching components. Valid range of credits is 1-127 credits, in increments of the single credit. 127 - indicates infinite credits
	5	TC1 Bandwidth Credits	
	6	TC2 Bandwidth Credits	
	7	TC3 Bandwidth Credits	
	8	TC4 Bandwidth Credits	
	9	TC5 Bandwidth Credits	
	10	TC6 Bandwidth Credits	
	11	TC7 Bandwidth Credits	
Reserved4	12-32	0	Reserved. Must be set to 0.

Table 7-159 lists the format of completion returned in the response buffer.



Table 7-159. Configure VSI Bandwidth Allocation per Traffic Type Completion Buffer

Category	Byte/Bit	Field	Description
QueueSet Handles	16-17	QS0 Handle	Handle per queue set. Completion will always carry eight queue set handles. Only TCs that were marked as a valid TCs in the Command will have a valid queue set handles returned in completion. Invalid handle is marked by 0xFFFF. Order of queue set handles matches order of TCs. Queue set handles must be used by software to associated transmit queue with queue set.
	18-19	QS1 Handle	
	20-21	QS2 Handle	
	22-23	QS3 Handle	
	24-25	QS4 Handle	
	26-27	QS5 Handle	
	28-29	QS6 Handle	
	30-31	QS7 Handle	

7.8.4.9 Configure Switching Component Bandwidth Limit

This command allows software to enable, disable or modify a bandwidth limit for the specified switch component. By enabling the bandwidth limit of a switching component, software is effectively enabling a bandwidth limit on the egress port of the switching component.

This command allows software to enable a global bandwidth limit regardless of the traffic type. To enable bandwidth limit per traffic type, software should use the commands described in [Section 7.8.4.13](#).

Software cannot have both bandwidth limit and bandwidth limit per traffic type enabled for the same switching component. A request to enable bandwidth limit for switching component that has bandwidth limit per traffic type enabled should fail, and an EPERM error should be reported in completion.

Under MFP, EMP firmware will ignore the TC Enable Bandwidth parameters of the AQC. Instead, EMP firmware will retrieve the parameters of TC enable and bandwidth configuration per PF from MFP Switch configuration in Alternate RAM data and UP to TC mapping.

In that case, if the PF uses more than one TC (such as iSCSI PF which must be opened for Storage TC and LAN TC as well), the following conditions apply:

- The PF is allowed to enable more than one TC, according to enabled UPs and UP to TC mapping.
- Bandwidth relative allocation defined for this PF in the Alternate Ram is configured for each one of the used TC's.
- Maximum bandwidth for the PF is declared as shared rate limiter.

The X710-TM4 provides hierarchical bandwidth limits. When configuring the bandwidth limit to the switching component, software automatically limits a total bandwidth available for all VSI allocated for that switching component. See [Section 7.8.1.3](#).

This is a direct Admin Queue command with completion reported within the completion descriptor structure. [Table 7-160](#) lists the command format and defines the command specific fields.

**Table 7-160. Configure Switching Component Bandwidth Limit Command Fields**

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0410	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Switching Component SEID	16-17		Unique identifier of the switching component
Reserved1	18-19	0	Reserved. Must be set to 0.
Bandwidth Limit Credits	20-21		Bandwidth limit in Mb/s. Supported range is 50 Mb/s, in increments of 50 Mb/s. 0 indicates that bandwidth limit is disabled. One credit corresponds to bandwidth limit of 50 Mb/s.
Bandwidth Limit Max	24.0-24.2		Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of Quanta-worth credits: 0, 1, 2, and 4 Quanta worth credits.
Reserved2	24.3-31	0	Reserved. Must be set to 0.

7.8.4.10 Enable, Disable or Modify Physical Port ETS

This command allows software to enable, disable or modify ETS configuration of the specified switching component connected directly to the Physical Port. Configuring ETS for the switching component effectively results in configuring ETS for the switching component egress port.

This command can only be used for switching components directly connected to the physical port configured to the ETS-Based Scheduler configuration. Otherwise, the command is ignored, and an error is reported.

Software is expected to store a current ETS configuration of the switching component, and fully specify ETS configuration every time it requests its modification. Software cannot provide incremental changes to the ETS configuration.

Firmware will configure TC arbitration to be a weighted round robin arbitration scheme. Software can choose to specify a different arbitration scheme, such as strict priority or a combination of weighted and robin and weighted strict priority. Software also has the option to provide an infinite number of credits to strict priority TCs.

This is an indirect Admin Queue command with additional command attributes. The completion attributes are provided within the data buffer. [Table 7-161](#) lists the command format and its command-specific fields.



Table 7-161. Configure Physical Port ETS Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0413 0x0414 0x0415	Command opcode 0x0413 - Enable ETS 0x0414 - Modify ETS 0x0415 - Disable ETS
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Physical Port SEID	16-17		Unique identifier of the physical port
Reserved	18-23	0	Reserved. Must be set to 0.
Data Address high	24-17		Address of buffer.
Data Address low	28-31		

Table 7-162 lists the format of the data buffer carrying additional command attributes. The majority of the fields in this table are valid for ETS enable and modify operations only. The ETS disable operation should refer to a Switching Component SEID only.

Table 7-162. Configure Physical Port ETS Command Buffer

Category	Byte/Bit	Field	Description
Reserved1	0-3		Reserved to match VSI configuration format
TC Valid	4.0-4.7	TC0-TC7 Valid	Valid bit per TC. Should list all TCs enabled for physical port. At list one TC must be enabled.
Enable seepage	5.0		When En see page is set to zero, Non Valid TC are suspended. when En see page is set, Non Valid TC are not suspended but will be configured with the lowest possible RL,
Reserved1	5.1-5.7		Reserved. Must be set to 0.
TC Strict Priority	6.0-6.7	TC0-TC7 Strict Priority Flag	Strict priority flag per TC. If set then TC should be arbitrated based on its priority. If software decides to configure one or more TCs to be a strict priority TC, then TC with higher TC number has a higher priority.
Reserved2	7.0-7.7		Reserved. Must be set to 0.
Reserved3	8-23	0	Reserved to match VSI configuration format



Table 7-162. Configure Physical Port ETS Command Buffer

Category	Byte/Bit	Field	Description
TC Bandwidth Share Credits	24	TC0 Bandwidth Credits	Relative credits per TC. Valid range of credits is 1-127 credits, in increments of the single credit. 127 - indicates infinite credits if TC is configured to strict priority
	25	TC1 Bandwidth Credits	
	26	TC2 Bandwidth Credits	
	27	TC3 Bandwidth Credits	
	28	TC4 Bandwidth Credits	
	29	TC5 Bandwidth Credits	
	30	TC6 Bandwidth Credits	
	31	TC7 Bandwidth Credits	
Reserved4	32-127		Reserved to match VSI configuration format

7.8.4.11 Suspend Port’s TX Traffic

This command enables the PF to suspend port's transmit traffic. The command is completed immediately. After all queue sets belonging to the port are suspended and the TX pipe of the port is drained, EMP firmware will use the Send Link Status event notification to update the PF. During the time period between processing this command and getting the event notification about port draining, the AQ commands Set PHY Config, Set MAC Config, and Set Link and Restart AN for this port, are forbidden and will be rejected with error code EAGAIN {8}.

Prior to calling this Admin command, the PF must use Set Event Mask to register itself to link events notifications.

This command enables software to modify a port's configuration (such as its DCB setting) when it owns the LLDP agent.

Port draining is part of the DCB configuration flow. This command is used to verify that the TX pipe is drained from any TX packet belonging to the port. This is a pre-condition to some configuration changes of the port TC or DCB. This command is valid only under SFP mode. See [Section 7.8.5.6.1](#) for more details.

This is a direct Admin Queue command with completion reported within the Direct Command Completion structure. [Table 7-163](#) lists command format and defines its specific fields.

Table 7-163. Suspend Port’s Tx Traffic

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x041B	Command opcode
Datalen	4-5	0	Length of response buffer
Return value	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Physical Port SEID	16-17		Unique identifier of the physical port
Reserved	18-31	0	Reserved. Must be set to 0.



7.8.4.12 Resume PF Traffic

This command is used to resume suspended TX traffic for the PF. It will resume all queue sets belonging to the PF, (all its VEBs and VSIs for all their TCs). This command is called by the PF driver after the completion of a configuration flow which requires the Port's TX pipe to be suspended. See [Section 7.8.5.6.1](#) for more details.

This is a direct Admin Queue command with completion reported within the direct command completion structure. [Table 7-163](#) lists the command format and its specific fields.

Table 7-164. Resume PF Traffic

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x041C	Command opcode
Datalen	4-5	0	Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31	0	Reserved. Must be set to 0.

7.8.4.13 Configure Switching Component Bandwidth Limit per Traffic Type

This command allows software to specify the TCs enabled for switching components, and to configure bandwidth limits of TCs enabled for the specified switch component.

Software should keep track of already configured bandwidth limits, and provide complete bandwidth limit configuration, not just incremental modifications.

This command can only be used for switching components belonging to the physical port that is configured to ETS-Based Scheduler configuration. Otherwise, the command is ignored, and an error is reported.

Software is allowed to change bandwidth limit per traffic type and a TC enabled for switching component that has a bandwidth allocation configured. A new TC enabled for a switching component is allocated a default bandwidth. Software should use a configure switching component Bandwidth Allocation per VSI AQ command to modify a default bandwidth allocation.

This command can be used to enable bandwidth limit per TC for the physical port.

Software cannot have both bandwidth limit and bandwidth limit per traffic type enabled for the same switching component. A request to enable bandwidth limit per traffic type for the switching component that has bandwidth limit enabled should fail and cause an EPERM error to be reported.

Software can configure either switching component bandwidth limit, or switching component TC bandwidth limit.

Under MFP, EMP firmware will ignore TC Enable Bandwidth parameters of the AQC. Instead, EMP firmware will retrieve the parameters of TC enable and bandwidth configuration per PF from the MFP Switch configuration in Alternate RAM data, along with the UP to TC mapping.



In that case, if the PF uses more than one TC (such as iSCSI PF which must be opened for Storage TC and LAN TC as well), the following conditions will be present:

- The PF is allowed to enable more than one TC. According to the enabled UPs and UP to TC mapping.
- Bandwidth-relative allocation defined for this PF in the Alternate Ram will be configured for each one of the used TCs.
- Maximum bandwidth for the PF will be declared as a shared rate limiter.

If TC re-configuration is needed while the VSI is suspended, newly generated TC nodes of this VSI will also be suspended.

This is an indirect Admin Queue command. Software should provide a valid buffer carrying additional command attributes as defined in Table 7-165. Command completion does not provide any additional information beyond status, and does not use a data buffer provided by software for command attributes.

Table 7-165 lists the command format and its command specific fields.

Table 7-165. Configure Switching Component Bandwidth Limits per Traffic Type Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0416	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Switching Component SEID	16-17		Unique identifier of the switching component or physical port.
Reserved	18-23	0	Reserved. Must be set to 0.
Data Address high	24-17		Address of buffer.
Data Address low	28-31		

Table 7-166 list the format of the data buffer carrying additional command attributes.

Table 7-166. Configure Switching Component Bandwidth Limit per Traffic Type Command Buffer

Category	Byte/Bit	Field	Description
TC Valid	0.0-0.7	TC0-TC7 Valid	Valid bit per TC. Should list all TCs enabled for switching component. Must be consistent with the physical port configuration. At least one TC must be enabled.
Reserved0	1-15	0	Reserved. Must be set to 0.



Table 7-166. Configure Switching Component Bandwidth Limit per Traffic Type Command Buffer

Category	Byte/Bit	Field	Description
TC Bandwidth Limit Credits	16-17	TC0 Bw Limit Credits	This field specifies a bandwidth limit per TC in Mb/s. Supported bandwidth limit range is 50 Mb/s. Bandwidth limit of 0Mb/s indicates that bandwidth limit is disabled. Bandwidth limit should be provided for the valid TC handles only. One credit corresponds to bandwidth limit of 50Mb/s
	18-19	TC1 Bw Limit Credits	
	20-21	TC2 Bw Limit Credits	
	22-23	TC3 Bw Limit Credits	
	24-25	TC4 Bw Limit Credits	
	26-27	TC5 Bw Limit Credits	
	28-29	TC6 Bw Limit Credits	
	30-31	TC7 Bw Limit Credits	
TC Max Bandwidth Limit	32.0-32.2	TC0 Max Bw Limit	Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of Quanta-worth credits: 0,1, 2, 4, 8,16 32 and 64 Quanta worth credits.
	32.3	Reserved	
	32.4-32.6	TC1 Max Bw Limit	
	32.7	Reserved	
	33.0-33.2	TC2 Max Bw Limit	
	33.3	Reserved	
	33.4-33.6	TC3 Max Bw Limit	
	33.7	Reserved	
	34.0-34.2	TC4 Max Bw Limit	
	34.3	Reserved	
	34.4-34.6	TC5 Max Bw Limit	
	34.7	Reserved	
	35.0-35.2	TC6 Max Bw Limit	
	35.3	Reserved	
	35.4-35.6	TC7 Max Bw Limit	
35.7	Reserved		
Reserved2	36-63	0	Reserved. Must be set to 0.

7.8.4.14 Configure Switching Component Bandwidth Allocation per Traffic Type

This command allows software to specify TCs enabled for a switching component, and to configure relative bandwidth allocation of switching components within each TC.

This command can be used for switching components configured in ETS-based mode only. Otherwise, the command would be ignored, and an error reported.

To configure the switching component bandwidth allocation per traffic type, software should provide complete relative bandwidth allocation for all TCs enabled for the switching component. In the MFP environment, when different switching components sharing the same physical port are owned by different physical functions, software is allowed to provide absolute bandwidth allocation credits, and firmware is responsible for translating those into relative credits.



Software is allowed to change bandwidth allocation per traffic type, and a TC enabled for switching components has a switching component bandwidth limit enabled. It the responsibility of firmware to adjust hardware configuration to reflect this change.

Under MFP, EMP firmware will ignore TC Enable Bandwidth parameters of the AQC. Instead, EMP firmware will retrieve the parameters of TC Enable and bandwidth configuration per PF from MFP Switch configuration in Alternate RAM data, and UP to TC mapping.

In that case, if the PF uses more than one TC (such as iSCSI PF which must be opened for Storage TC and LAN TC as well), the following conditions will be present:

- The PF is allowed to enable more than one TC, according to enabled UPs and UP to TC mapping.
- Bandwidth relative allocation defined for this PF in the Alternate Ram will be configured for each one of the used TCs.
- Maximum bandwidth for the PF will be declared as shared rate limiter.

If TC re-configuration is needed while the VSI is suspended, new generated TC nodes of this VSI will also be suspended.

This is an indirect Admin Queue command. Software should provide a buffer that would be used both to provide additional command attributes and for command completion.

Table 7-167 lists the command format and defines a command-specific fields.

Table 7-167. Configure Switching Component Bandwidth Allocation per Traffic Type Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0417	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Switching Component SEID	16-17		Unique identifier of the VSI
Reserved1	18-23	0	Reserved. Must be set to 0.
Data Address high	24-27		Address of buffer.
Data Address low	28-31		This buffer is used both to convey configuration to firmware, and provide software with a list of TC Handles.

Table 7-168 lists the format of the command buffer and defines command attributes.



Table 7-168. Configure Switching Component Bandwidth Allocation per Traffic Type Command Buffer

Category	Byte/Bit	Field	Description
TC Valid	0.0-0.7	TC0-TC7 Valid	Valid bit per TC. Should list all TCs enabled for switching component. Must be consistent with the physical port configuration. At least one TC must be enabled.
Reserved2	1-3.6	0	Reserved. Must be set to 0.
Absolute Credits	3.7	Absolute Credits Enable	If set indicates that credits provided are absolute credits, and not relative to the bandwidth allocated to other switching components on the same TC. Software should avoid use of absolute credits.
TC Bandwidth Share Credits	4	TC0 Bandwidth Credits	Relative switching component credits within same TC with respect to other switching components or VSIs enabled and connected to s-channels on the same physical port Valid range of credits is 1-127 credits, in increments of the single credit. 127 - indicates infinite credits
	5	TC1 Bandwidth Credits	
	6	TC2 Bandwidth Credits	
	7	TC3 Bandwidth Credits	
	8	TC4 Bandwidth Credits	
	9	TC5 Bandwidth Credits	
	10	TC6 Bandwidth Credits	
	11	TC7 Bandwidth Credits	
Reserved4	12-32	0	Reserved. Must be set to 0.

7.8.4.15 Query VSI Bandwidth Configuration

This command enables software retrieve current bandwidth configuration of the specified VSI of the specified switching component.

This is an indirect Admin Queue command. Software should provide a buffer for command completion. Command completion carries VSI bandwidth configuration attributes, listed in [Table 7-169](#).

DCB flow might require changes to both the VEB and VSI configurations. Calling Query VSI Bandwidth Configuration between the previous steps will result in an error code EBUSY (12).

[Table 7-169](#) lists a command format, and defines a command specific fields.

Table 7-169. Query VSI Bandwidth Configuration Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0408	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. This should carry a status and an indication whether ETS is enabled or not.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
VSI SEID	16-17		Unique identifier of the VSI



Table 7-169. Query VSI Bandwidth Configuration Command Fields

Name	Bytes.Bits	Value	Remarks
Reserved	18-23	0	Reserved. Must be set to 0.
Data Address high	24-27		Address of buffer.
Data Address low	28-31		

Table 7-170 lists the format of the completion buffer, and defines completion attributes. In addition to the bandwidth characteristics of VSI, this completion carries UP Handles and QS Handles of UPs and queue sets allocated for VSI.

Table 7-170. Query VSI Configuration Completion Buffer

Category	Byte/Bit	Field	Description
TC Valid	0.0-0.7	TC0-TC7 Valid	Valid bit per TC.
TC Suspended	1.0-1.7	TC0-TC7 Suspended	Marks per TC if this TC transmission is suspended.
Reserved0	2-15	0	Reserved. Must be set to 0.
QueueSet Handles	16-17	QS0 Handle	QueueSet handles of configured queue sets. Invalid handles will carry value of 0xffff. Order of queue set handles matches order of TCs (0-7).
	18-19	QS1 Handle	
	20-21	QS2 Handle	
	22-23	QS3 Handle	
	24-25	QS4 Handle	
	26-27	QS5 Handle	
	28-29	QS6 Handle	
	30-31	QS7 Handle	
Reserved1	32-35	0	Reserved. Must be set to 0.
Bandwidth Limit	36-37		Bandwidth limit configured for the VSI in Mb/s. 0 indicates that bandwidth limit was not configured for the VSI One credit corresponds to bandwidth limit of 50Mb/s
Reserved2	38-39	0	Reserved. Must be set to 0.
Bandwidth Limit Max	40.0-40.2		Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of Quanta-worth credits: 0,1, 2, and 4 Quanta worth credits.
Reserved3	40.3-63	0	Reserved. Must be set to 0.

7.8.4.16 Query VSI Bandwidth Configuration per Traffic Type

This command enables software to retrieve a bandwidth configuration of the specified VSI within each TC.

This command can be used for VSI configured in ETS-Based mode only. Otherwise, the command is ignored, and an error is reported.



If the Physical Port ETS is configured to a single level ETS, it defines bandwidth distribution between TCs only. Software can assume one-to-one mapping of UPs to TCs from the bandwidth allocation perspective, and can use this command to retrieve a bandwidth allocation for VSI within each TC.

This is an indirect Admin Queue command. Software should provide a buffer for additional command attributes and command completion. Command completion carries VSI ETS/SLA configuration attributes, listed in [Table 7-171](#).

DCB flow might require changes to both the VEB and VSI configurations. Calling Query VSI Bandwidth Configuration between these two steps will result in an error code EBUSY (12).

[Table 7-171](#) lists the command format, and the command specific fields.

Table 7-171. Query VSI Bandwidth Allocation per Traffic Type Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x040A	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. This should carry a status and an indication whether ETS is enabled or not.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
VSI SEID	16-17		Unique identifier of the VSI
Reserved	18-23	0	Reserved. Must be set to 0.
Data Address high	24-27		Address of buffer.
Data Address low	28-31		

[Table 7-172](#) lists the format of completion buffer, and its attributes.

Table 7-172. Query VSI Bandwidth Allocation per Traffic Type Completion Buffer

Category	Byte/Bit	Field	Description
TC Valid	0.0-0.7	TC0-TC7 Valid	Valid bit per TC
TC Suspended	1.0-1.7	TC0-TC7 Suspended	Marks per TC if this TC transmission is suspended.
Reserved1	2-3	0	Reserved. Must be set to 0.
TC Bandwidth Share Credits	4	TC0 Bandwidth Credits	Relative VSI credits within same TC with respect to other VSIs enabled on the same switching component Valid range of credits is 1-127 credits, in increments of the single credit. 127 - indicates infinite credits
	5	TC1 Bandwidth Credits	
	6	TC2 Bandwidth Credits	
	7	TC3 Bandwidth Credits	
	8	TC4 Bandwidth Credits	
	9	TC5 Bandwidth Credits	
	10	TC6 Bandwidth Credits	
	11	TC7 Bandwidth Credits	



Table 7-172. Query VSI Bandwidth Allocation per Traffic Type Completion Buffer

Category	Byte/Bit	Field	Description
TC Bandwidth Limit Credits	12-13	TC0 Bw Limit Credits	This field specifies a bandwidth limit per TC in Mb/s. Supported bandwidth limit range is 50 Mb/s. Bandwidth limit of 0Mb/s indicates that bandwidth limit is disabled. Bandwidth limit should be provided for the valid handles only. One credit corresponds to bandwidth limit of 50 Mb/s
	14-15	TC1 Bw Limit Credits	
	16-17	TC2 Bw Limit Credits	
	18-19	TC3 Bw Limit Credits	
	20-21	TC4 Bw Limit Credits	
	22-23	TC5 Bw Limit Credits	
	24-25	TC6 Bw Limit Credits	
	26-27	TC7 Bw Limit Credits	
TC Max Bandwidth Limit	28.0-28.2	TC0 Max Bw Limit	Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of Quanta-worth credits: 0,1, 2, 4, 8,16 32 and 64 Quanta worth credits.
	28.3	Reserved	
	28.4-28.6	TC1 Max Bw Limit	
	28.7	Reserved	
	29.0-29.2	TC2 Max Bw Limit	
	29.3	Reserved	
	29.4-29.6	TC3 Max Bw Limit	
	29.7	Reserved	
	30.0-30.2	TC4 Max Bw Limit	
	30.3	Reserved	
	30.4-30.6	TC5 Max Bw Limit	
	30.7	Reserved	
	31.0-31.2	TC6 Max Bw Limit	
	31.3	Reserved	
	31.4-31.6	TC7 Max Bw Limit	
31.7	Reserved		

7.8.4.17 Query Switching Component Configuration

This command enables software to retrieve current bandwidth management configuration of the specified switching component.

This is an indirect Admin Queue command. Software should provide a buffer for command completion. Command completion carries switching component bandwidth configuration attributes, listed in [Table 7-173](#).

[Table 7-173](#) lists a command format, and defines command specific fields.

The result of this command is returned as a completion to the Admin Queue command.



Table 7-173. Query Switching Component Configuration Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0418	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. This should carry a status and an indication whether ETS is enabled or not.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Switching Component SEID	16-17		Unique identifier of the switching component
Reserved	18-23	0	Reserved. Must be set to 0.
Data Address high	24-27		Address of buffer.
Data Address low	28-31		

Table 7-174 lists the format of a completion buffer and its attributes.

Table 7-174. Query Switching Component Configuration Completion Buffer

Category	Byte/Bit	Field	Description
TC Valid	0.0-0.7	TC0-TC7 Valid	Valid bit per TC
Reserved1	1-31	0	Reserved. Must be set to 0.
Reserved1	32-35	0	Reserved. Must be set to 0.
Bandwidth Limit	36-37		Bandwidth limit configured for the port in Mb/s. 0 indicates that bandwidth limit was not configured for the switching component One credit corresponds to bandwidth limit of 50 Mb/s
Reserved2	38-39	0	Reserved. Must be set to 0.
Bandwidth Limit Max	40.0-40.2		Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of Quanta-worth credits: 0,1, 2, and 4 Quanta worth credits.
Reserved3	40.3-63	0	Reserved. Must be set to 0.

7.8.4.18 Query Physical Port ETS Configuration

This command enables software to retrieve the ETS configuration of the specified switching component or the VSI directly connected to the Physical Port.

This is an indirect Admin Queue command. Software should provide a buffer for additional command attributes and command completion. Command completion carries switching component ETS/SLA configuration attributes, listed in Table 7-175.



This command can be used only for switching components directly connected to the physical port configured to an ETS-Based Scheduler configuration. Otherwise, the command is ignored, and error is reported.

Table 7-175 lists the command format and the command specific fields.

Table 7-175. Query Physical Port ETS Configuration Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x0419	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. This should carry a status and an indication whether ETS is enabled or not.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Physical Port SEID	16-17		Unique identifier of the physical port
Reserved	18-23	0	Reserved. Must be set to 0.
Data Address high	24-27		Address of buffer.
Data Address low	28-31		

Table 7-176 lists the format of a completion buffer, and its attributes.

Table 7-176. Query Physical Port ETS/SLA Completion Buffer

Category	Byte/Bit	Field	Description
Reserved1	0-3		Reserved to match VSI configuration format
TC Valid	4.0-4.7	TC0-TC7 Valid	Valid bit per TC This field is valid for ETS enable operation only. For ETS Modify operation TC Handles must be used instead Upto 8 TCs can be valid at any time.
TC Strict Priority	6.0-6.7	TC0-TC7 Strict Priority Flag	Strict priority flag per TC. If set then TC should be arbitrated based on its priority. If software decides to configure one or more TCs to be a strict priority TC, then TC with higher TC number has a higher priority. Configuring TCs to the strict priority might be useful for SLA configuration.
TC Bandwidth Share Credits	8	TC0 Bandwidth Credits	Relative credits per TC. Valid range of credits is 1-127 credits, in increments of the single credit. 127 - indicates infinite credits if TC is configured to strict priority
	9	TC1 Bandwidth Credits	
	10	TC2 Bandwidth Credits	
	11	TC3 Bandwidth Credits	
	12	TC4 Bandwidth Credits	
	13	TC5 Bandwidth Credits	
	14	TC6 Bandwidth Credits	
	15	TC7 Bandwidth Credits	



Table 7-176. Query Physical Port ETS/SLA Completion Buffer

Category	Byte/Bit	Field	Description
TC Bandwidth Limit Credits	16-17	TC0 Bw Limit Credits	This field specifies a bandwidth limit per TC in Mb/s. Supported bandwidth limit range is 50 Mb/s. Bandwidth limit of 0 Mb/s indicates that bandwidth limit is disabled. Bandwidth limit should be provided for the valid TC handles only. One credit corresponds to bandwidth limit of 50 Mb/s
	18-19	TC1 Bw Limit Credits	
	20-21	TC2 Bw Limit Credits	
	22-23	TC3 Bw Limit Credits	
	24-25	TC4 Bw Limit Credits	
	26-27	TC5 Bw Limit Credits	
	28-29	TC6 Bw Limit Credits	
	30-31	TC7 Bw Limit Credits	
TC Max Bandwidth Limit	32.0-32.2	TC0 Max Bw Limit	Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of Quanta-worth credits: 0,1, 2, 4, 8,16 32 and 64 Quanta worth credits.
	32.3	Reserved	
	32.4-32.6	TC1 Max Bw Limit	
	32.7	Reserved	
	33.0-33.2	TC2 Max Bw Limit	
	33.3	Reserved	
	33.4-33.6	TC3 Max Bw Limit	
	33.7	Reserved	
	34.0-34.2	TC4 Max Bw Limit	
	34.3	Reserved	
	34.4-34.6	TC5 Max Bw Limit	
	34.7	Reserved	
	35.0-35.2	TC6 Max Bw Limit	
	35.3	Reserved	
35.4-35.6	TC7 Max Bw Limit		
35.7	Reserved		
Reserved3	36-67	0	Reserved to match VSI configuration format. Must be set to 0.

7.8.4.19 Query Switching Component Bandwidth Configuration per Traffic Type

This command enables software to retrieve a bandwidth configuration of the specified switching component within each TC.

This command can be used for switching components configured in ETS-Based mode only. Otherwise command is ignored, and an error is reported.

This is an indirect Admin Queue command. Software should provide a buffer for additional command attributes and command completion. Command completion carries VSI ETS/SLA configuration attributes, listed in [Table 7-177](#).

[Table 7-177](#) lists the command format and defines command specific fields.



Table 7-177. Query Switching Component Bandwidth Allocation per Traffic Type Command Fields

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x041A	Command opcode
Datalen	4-5		Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. This should carry a status and an indication whether ETS is enabled or not.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
SEID	16-17		Unique identifier of the VSI or switching component
Reserved	18-23	0	Reserved. Must be set to 0.
Data Address high	24-27		Address of buffer.
Data Address low	28-31		

Table 7-178 lists the format of a completion buffer and its attributes.

Table 7-178. Query Switching Component Bandwidth Allocation per Traffic Type Completion Buffer

Category	Byte/Bit	Field	Description
TC Valid	0.0-0.7	TC0-TC7 Valid	Valid bit per TC
Reserved1	1-3.6	0	Reserved. Must be set to 0.
Absolute Credits	3.7	Absolute Credits Enable	If set indicates that credits provided are absolute credits, and not relative to the bandwidth allocated to other switching components on the same TC. Software should avoid use of absolute credits.
TC Bandwidth Share Credits	4	TC0 Bandwidth Credits	Relative switching component credits within same TC with respect to other switching components enabled on the same Physical Port Valid range of credits is 1-127 credits, in increments of the single credit. 127 - indicates infinite credits
	5	TC1 Bandwidth Credits	
	6	TC2 Bandwidth Credits	
	7	TC3 Bandwidth Credits	
	8	TC4 Bandwidth Credits	
	9	TC5 Bandwidth Credits	
	10	TC6 Bandwidth Credits	
TC Bandwidth Limit Credits	11	TC7 Bandwidth Credits	
	12-13	TC0 Bw Limit Credits	This fiend specifies a bandwidth limit per TC in Mb/s. Supported bandwidth limit range is 50 Mb/s. Bandwidth limit of 0Mb/s indicates that bandwidth limit is disabled. Bandwidth limit should be provided for the valid handles only. One credit corresponds to bandwidth limit of 50 Mb/s
	14-15	TC1 Bw Limit Credits	
	16-17	TC2 Bw Limit Credits	
	18-19	TC3 Bw Limit Credits	
	20-21	TC4 Bw Limit Credits	
	22-23	TC5 Bw Limit Credits	
24-25	TC6 Bw Limit Credits		
26-27	TC7 Bw Limit Credits		



Table 7-178. Query Switching Component Bandwidth Allocation per Traffic Type Completion Buffer

Category	Byte/Bit	Field	Description
TC Max Bandwidth Limit	28.0-28.2	TC0 Max Bw Limit	Maximum bandwidth limit indicates how much bandwidth limit credits can be accumulated due to inactivity. Single scheduling Quanta consumes multiple bandwidth limit credits. The X710-TM4 allows accumulating of following discrete values of Quanta-worth credits: 0,1, 2, 4, 8,16 32 and 64 Quanta worth credits.
	28.3	Reserved	
	28.4-28.6	TC1 Max Bw Limit	
	28.7	Reserved	
	29.0-29.2	TC2 Max Bw Limit	
	29.3	Reserved	
	29.4-29.6	TC3 Max Bw Limit	
	29.7	Reserved	
	30.0-30.2	TC4 Max Bw Limit	
	30.3	Reserved	
	30.4-30.6	TC5 Max Bw Limit	
	30.7	Reserved	
	31.0-31.2	TC6 Max Bw Limit	
	31.3	Reserved	
31.4-31.6	TC7 Max Bw Limit		
31.7	Reserved		

7.8.4.20 Configure Tx Port Settings

This command allows a PF to enable or disable a port's Tx settings. This is a direct command.

Firmware writes back the status of the command to the Return Value field. The firmware also updates the current property settings in the response.

When no property settings are requested (bits cleared), no configuration changes will be done by firmware and the current configuration settings are posted in the command response.

Table 7-179. Configure Tx Port Settings Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x041E	Command opcode
Datalen	4-5	0	Length of response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Port SEID	16-17		Physical Port's SEI



Table 7-179. Configure Tx Port Settings Command

Name	Bytes.Bits	Value	Remarks
Property	18-23	18.0: Reserved 18.1: Multi Qset mode enable 18.2: Multi-Qset mode disable 18.3-23.7 - Reserved	Tx scheduler Property being modified Multi-Qset mode enable or disable are mutually exclusive settings and setting both bits to one is illegal and will be responded with and error EINVAL
Data Address high	24-27	0	Address of buffer.
Data Address low	28-31	0	

Table 7-180. Configure Tx Port Settings Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0	0	See Section 7.10.5 for details.
Opcode	2-3	0x041E	Command opcode
Datalen	4-5	0	Length of response buffer
Return value/VFID	6-7		Return Value. Zeroed by the software device driver. Written by firmware. Status of request. A value of SUCCESS means the command was performed successfully. Error codes: EINVAL - If invalid property value EMODE - MFP mode
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Reserved	16-17		
Property	18-23	18.0: Reserved 18.1: Multi-Qset mode enabled 18.2: Multi-Qset mode disabled 18.3-23.7 - Reserved	Tx scheduler property current setting
Reserved	24-31		Address of buffer.

7.8.5 Scheduler Configuration Flows

This section describes use of AdminQ commands defined in [Section 7.8.4](#) to configure the scheduler. It is not intended to provide complete coverage of all possible configuration flows, but to cover several main stream cases.

[Table 7-180](#) lists all possible configuration changes of the Internal Switch or bandwidth configuration that have an impact on the transmit scheduler configuration, along with AQ Commands that should be used.



Table 7-181. Scheduler Configuration Summary

Configuration Change	Description	AQ Command
Default Port ETS Configuration	Default ETS configuration is ETS disabled, and the only TC enabled is TC0. See Section 7.8.5.3 .	NA
Disable Physical Port	Disabling previously configured physical port does not directly impact scheduler configuration. Firmware will suspend disabled port, and software is responsible for resource deallocation using Delete Element AQ Command.	NA
Change Physical Port ETS Configuration	Driven by agent implementing DCBX protocol. Agent can run in firmware or in software. In case of software AQ command should be used to perform required ETS configuration change. See Section 7.8.5.3 .	Enable, Disable, Modify Physical Port ETS. Section 7.8.4.10
Add Default VSI	Default VSI is allocated at switch initialization time. SFP configuration - one default VSI per Port. MFP configuration - one default VSI per PF. Default VSI is configured with default bandwidth configuration (even bandwidth allocation, no bandwidth limits), and with single TC enabled - TC0. See Section 7.8.5.5.1 .	NA
Add VSI	Regular VSI can be added directly connected to the Physical Port, or VEB. Scheduler configuration is done as a part of the Internal Switch configuration. New VSI is configured with a default bandwidth configuration (single bandwidth allocation credit, and no bandwidth limit enabled). New VSI can be configured with an initial set of TCs enabled for VSI. TCs enabled for VSI must be a subset of TCs enabled to the parent switching element or Physical Port. Section 7.8.5.3 .	Add VSI Section 7.4.9.5.5.1 ,
Change TCs enabled for Default VSI or regular VSI.	Default or regular VSI TC configuration can be changed using Configure VSI Bandwidth Allocation per Traffic Type or Configure VSI Bandwidth Limit per Traffic Type AQ commands. In either case TCs enabled for the default VSI must be a subset of TCs enabled for the corresponding Physical Port. Update VSI cannot be used to change TC configuration of transmit scheduler.	Configure VSI Bandwidth Allocation per Traffic Type. Section 7.8.4.8 . Configure VSI Bandwidth Limit per Traffic Type. Section 7.8.4.7
Change VSI bandwidth configuration	To change VSI bandwidth configuration software can use one of three AQ Commands: Configure VSI Bandwidth Limit - to enable/modify/disable bandwidth limit that applies to entire VSI (all enabled TCs) Configure VSI Bandwidth Limit per Traffic Type - to enable/modify/disable bandwidth limit for individual TC enabled for VSI. Configure VSI Bandwidth Allocation per Traffic Type - to modify bandwidth allocation for the individual TC enabled for VSI.	Configure VSI Bandwidth Limit, Section 7.8.4.5 Configure VSI Bandwidth Allocation per Traffic Type. Section 7.8.4.8 . Configure VSI Bandwidth Limit per Traffic Type. Section 7.8.4.7
Delete VSI	Software can remove VSI using Delete Element AQ Command. This includes removing all TC Nodes enabled for VSI.	Delete Element, Section 7.4.9.5.8.1
Add VEB to Default VSI	When VEB is added to the Default VSI, it is effectively inserted between default VSI and Physical Port/ Port Virtualizer . VEB can be configured with a set of TCs enabled for VEB. TCs enabled for VEB must include all TCs enabled for the Default VSI. VEB inherits bandwidth configuration of Default VSI. TCs that are enabled for VEB and not enabled for the Default VSI must be configured with a default bandwidth configuration. See Section 7.8.5.2 .	Add VEB/Port Virtualizer. Section 7.4.9.5.6.1 , Section 7.4.9.5.7.1
Add floating VEB	Not supported by current Scheduler Definition. Transmit scheduler must provide a Physical Port and TC along with the scheduling request to allow proper chip resource allocation by the pipeline. There are several ways how floating VEB can be enabled for the scheduler. All options result in modification of the Scheduler configuration tables.	Add VEB/PA. Section 7.4.9.5.6.1 , Section 7.4.9.5.7.1



Table 7-181. Scheduler Configuration Summary

Configuration Change	Description	AQ Command
Change TCs enabled for VEB	To change number of TCs enabled for VEB software should use either Configure switching component Bandwidth Limit per Traffic Type, or Configure Bandwidth Allocation per Traffic Type AQ commands. TCs enabled for VEB must be a subset of TCs enabled for the parent switching component or Physical Port. Update VEB CANNOT be used to change TC configuration of transmit scheduler.	Configure switching component Bandwidth Limit per Traffic Type. Section 7.8.4.13 Configure switching component Bandwidth Allocation per Traffic Type. Section 7.8.4.13
Change VEB bandwidth configuration.	To change VEB bandwidth configuration software can use one of three AQ Commands: Configure switching component Bandwidth Limit - to enable/modify/disable bandwidth limit that applies to entire VEB (all enabled TCs) Configure switching component Bandwidth Limit per Traffic Type - to enable/modify/disable bandwidth limit for individual TC enabled for VEB. Configure switching component Bandwidth Allocation per Traffic Type - to modify bandwidth allocation for the individual TC enabled for VEB.	Configure switching component Bandwidth Limit. Section 7.8.4.9. Configure switching component Bandwidth Limit per Traffic Type. Section 7.8.4.14 Configure switching component Bandwidth Allocation per Traffic Type. Section 7.8.4.13
Delete VEB	Software can remove VEB using Delete Element AQ Command. This includes removing all TC Nodes enabled for VEB. Software can remove VEB only after removing all VSIs attached to it after VEB creation. When VEB is removed, a default VSI is attached back to the S-Channel or Physical Port.	Delete Element, Section 7.4.9.5.8.1

7.8.5.1 Default VSI

A default VSI is automatically created for each physical function. In an SFP environment, a single default VSI is allocated per physical port. In an MFP environment, multiple default VSIs can be created per physical port - one per physical function. The default VSI is always associated with TC0. The default VSI owns all bandwidth allocated for the physical port or physical function. Software can change the TCs enabled for the default VSI and bandwidth allocation per traffic type using AQ command described in [Section 7.8.4.8](#).

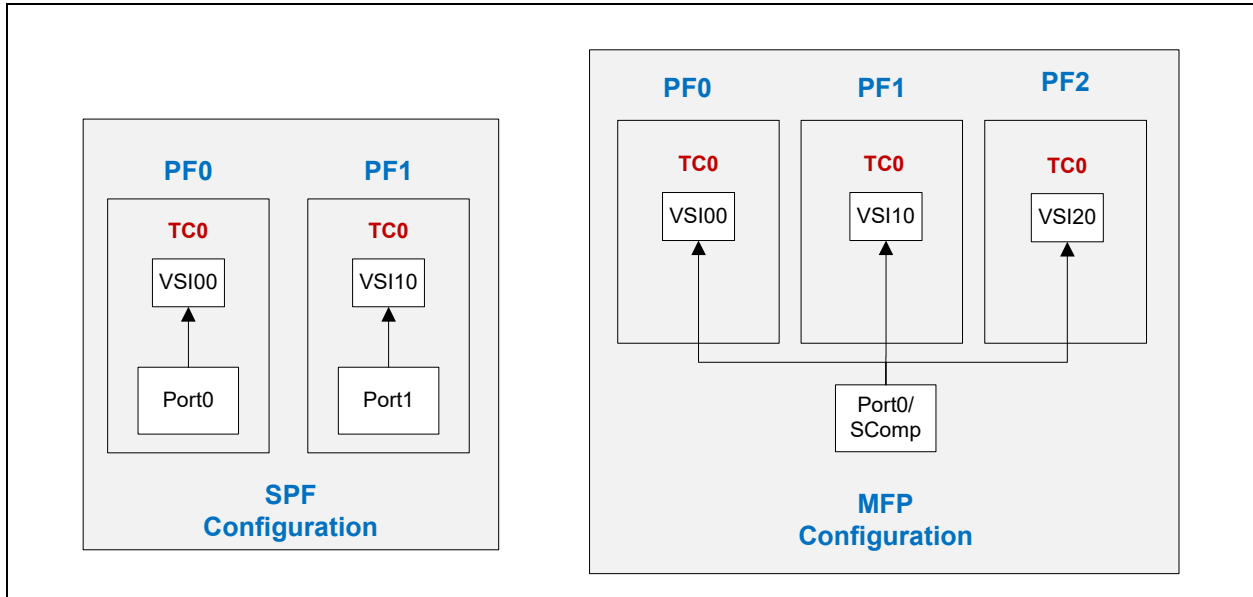


Figure 7-73. Default VSI Configuration

Figure 7-73 shows an example of the default VSI configuration in SFP and MFP modes.

Firmware allocates a queue set for the each default VSI. Software can retrieve a queue set Handle using the Query VSI command. No additional configuration of default VSI is required, and once the respective queue set handle is associated with transmit queue, that queue can be scheduled for transmission.

ETS is disabled for the default VSI. Software can enable ETS configuration for the physical port and then configure ETS for VSI. See [Section 7.8.5.3](#).

Software can instantiate a switching component using the Add VEB or Add Port Virtualizer command, as described in [Section 7.8.5.2](#).

7.8.5.2 Adding VEB/PA/PE

Adding S-comp (port extender) does not impact transmit scheduler configuration. S-comp effectively replaces a physical port in the transmit scheduler configuration hierarchy. All attributes configured for the port automatically apply for the S-comp (ETS configuration, for instance.)

VEB/PA added to the switching hierarchy can be inserted between a default VSI and a physical port or an S-comp, or can be instantiated using its own S-channel.

Regular VEB, attached to physical port, is always inserted between a port or port extender and its VSI. If the VSI is not a default VSI automatically allocated by firmware for each physical function or physical port, software must allocate the VSI prior to adding VEB.

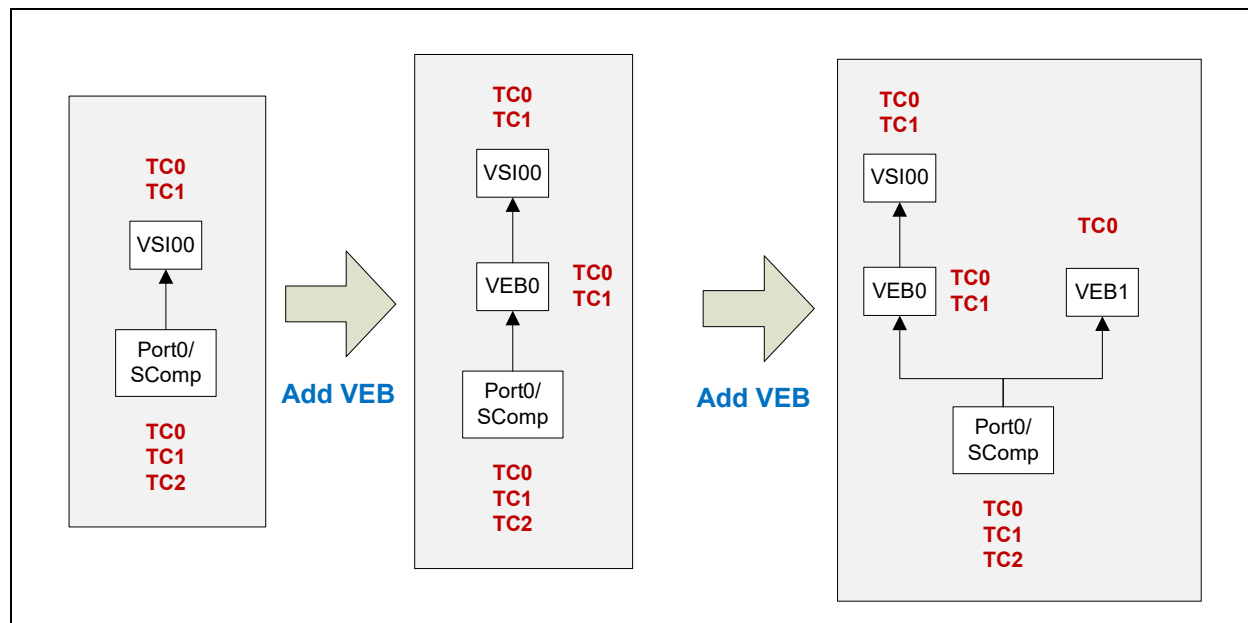


Figure 7-74. Adding VEB/PA

Figure 7-74 shows the steps to add a VEB/PA switching component.

First step shows the insertion of the VEB between the physical port and a default VSI (VSI00). In this example, the port has an ETS enabled, and a default VSI is configured with multiple TCs. The inserted VEB inherits configuration from the VSI, (the TCs enabled for the VSI, and the bandwidth management attributes). The VEB might have more TCs enabled than the VSI. Software can change TCs enabled for VEB and relative bandwidth allocation within each TC using the AQ command described in [Section 7.8.4.14](#).

The insertion of a VEB between the default VSI and the S-channel does not impact queue set allocated for VSI. Transmit queues, if any, previously associated with queue sets allocated for Default VSI will remain operational.

The second step shows a new VSI directly attached to the S-channel. Software is required to add VSI prior to adding a new VEB. In this example, the new VSI has TC0 enabled. Software can specify TCs enabled for VSI using a TC enable bitmap in the Add VSI AQ Command, or can change it later using Configure VSI Bandwidth Allocation per Traffic Type, or Configure VSI Bandwidth Limit per Traffic Type AQ commands. Upon allocation of VSI, firmware allocates a queue set for each TC enabled for VSI, and provides it in the AQ Command completion.

The third step shows a second VEB/PA inserted between VSI10 the port extender (VEB1). The process of creation and bandwidth configuration is similar to one described for the VEB0.

7.8.5.3 Adding a VSI

In addition to the default VSI created for each port in SFP mode and for each PF in MFP mode, software can request adding a VSI to the previously allocated switching component. By default, VSI is allocated with a single TC enabled - TC0. Software can configure TCs enabled for VSI either using a bit mask in the Add VSI AQ command, or using the scheduler configuration AQ command described in [Section 7.8.4.8](#).

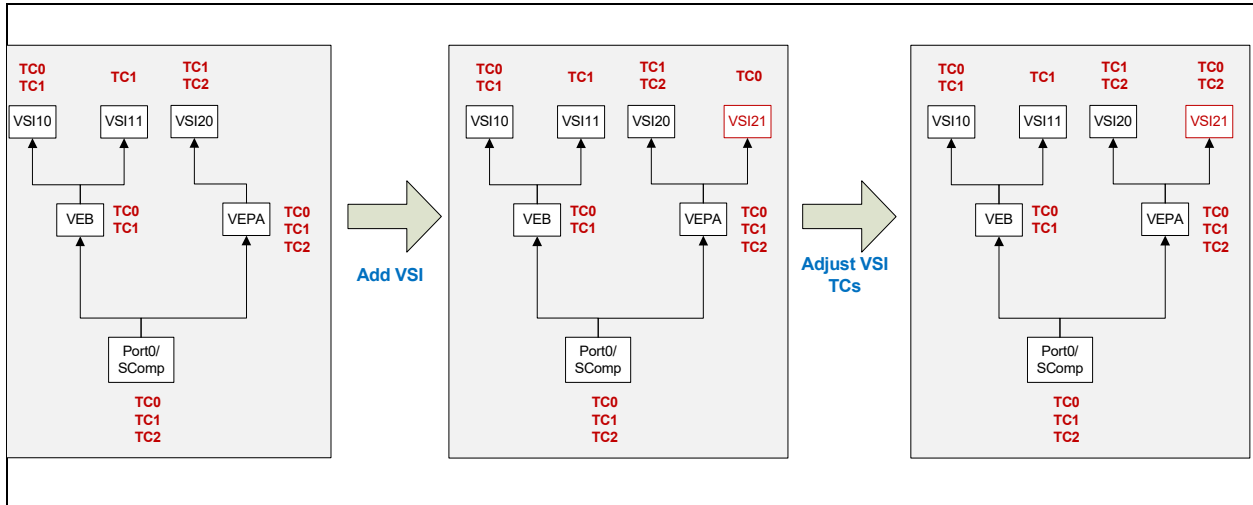


Figure 7-75. Add VSI

Each allocated VSI is allocated a queue set per enabled TC. Queue set handles are returned in completion of the Add VSI, or the AQ command described in [Section 7.8.4.8](#).

If the VSI is added to a port which is configured with Multi_Qset mode enabled, then each allocated VSI is allocated eight contiguous queue sets per enabled TC. The first queue set handle per each TC is returned in completion of the Add VSI command. Software is aware of the Port's Multi_Qset configuration and might associate TX queues with all eight queue sets.

queue sets.

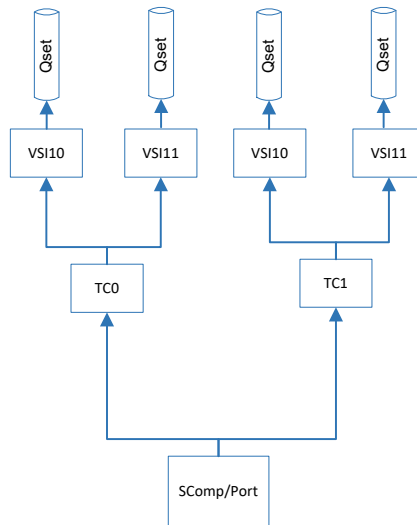


Figure 7-76. Port's Topology When Not Configured for Multi Qset

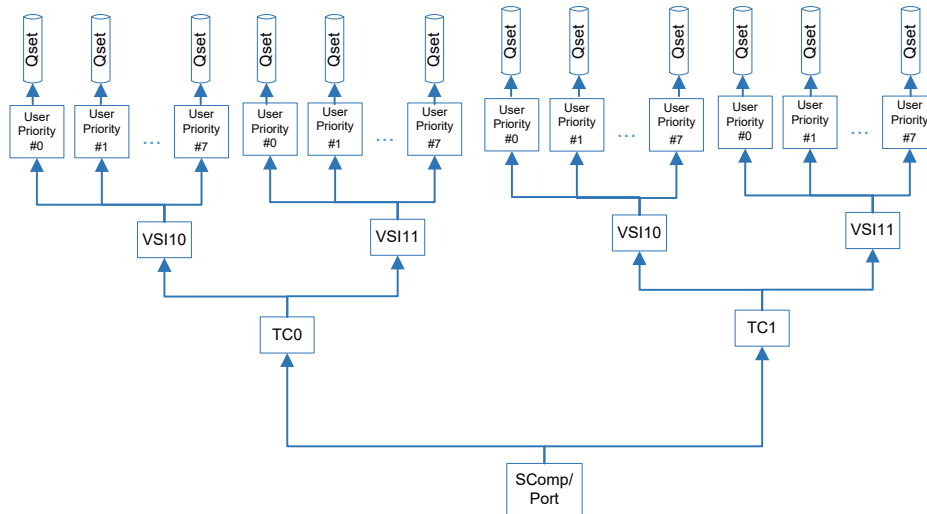


Figure 7-77. Port Topology When Configured for Multi Qset

7.8.5.4 Bandwidth Limiting

Each allocated switching component and VSI might have a bandwidth limit configured. The X710-TM4 supports two kinds of bandwidth limits - global, including traffic generated on all TCs enabled for VSI or switching component, and per traffic type bandwidth limit. Global bandwidth limits can be enabled using AQ commands described in [Section 7.8.4.9](#) and [Section 7.8.4.5](#). Per TC bandwidth limits can be enabled using AQ commands described in [Section 7.8.4.7](#).

Bandwidth limits are by default disabled. Software can enable only one kind of bandwidth limit to each switching component and VSI.

Software does not have to modify relative bandwidth allocation for the switching components and VSIs to enable bandwidth limits. The default (even) bandwidth allocation can be used, and bandwidth would be equally distributed between switching components and VSIs as long as they have not reached their configured bandwidth limit.

7.8.5.5 Bandwidth Distribution Ownership

7.8.5.5.1 SFP Mode

In SFP configuration, one physical function owns all resources allocated to the physical port.

ETS configuration of the physical port by default would be performed by firmware based on data provided by DCBX agent running in firmware. Software can take over the role of DCBX agent, and then it will control ETS configuration of the physical port as well. Refer to [Section 7.7.3](#) for details on DCBX ownership transfer. In either case, allocation or bandwidth management of the switching components and VSIs is completely owned by software, and performed using AQ commands described in [Section 7.8.4](#).

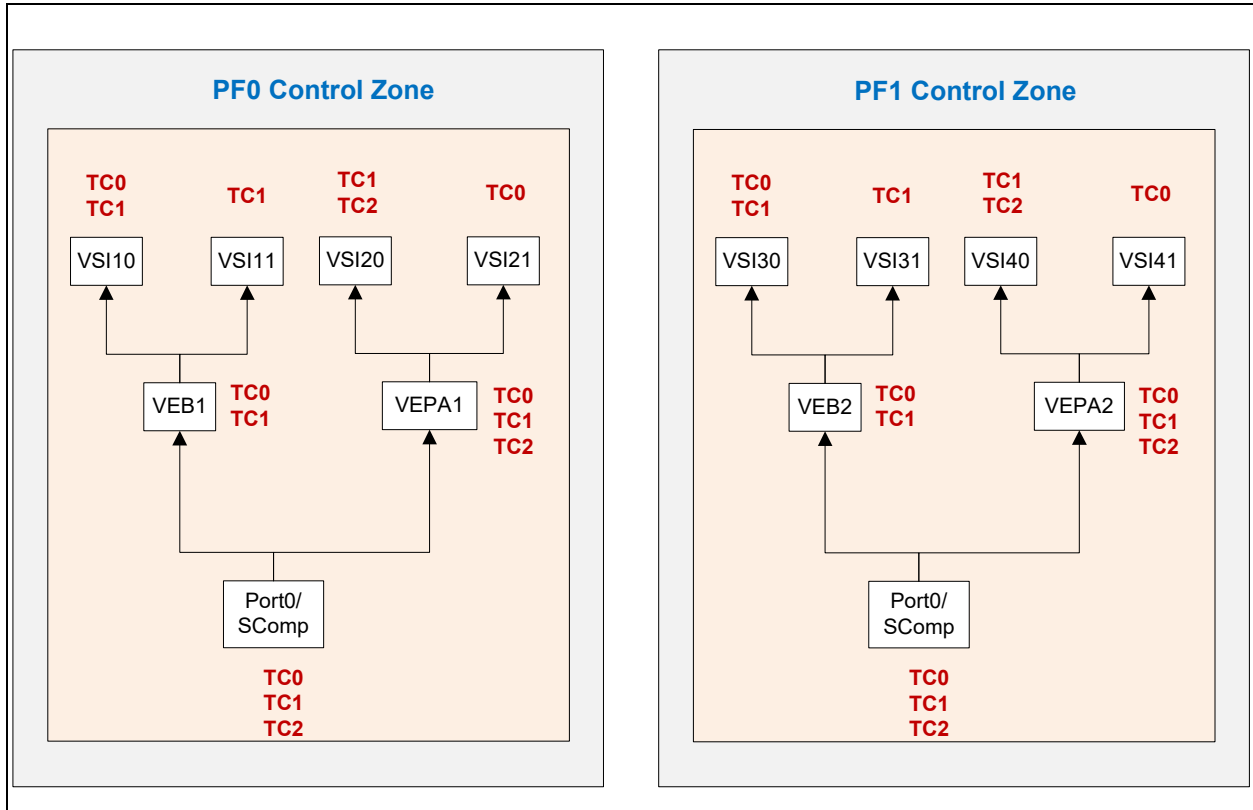


Figure 7-78. Software Control Zone in SFP Mode

Bandwidth allocation should be done using relative credits only. The PF driver should have all information available to manage relative credits.

7.8.5.5.2 MFP Mode

In MFP configuration, multiple PFs share a physical port. The allocation of resources between PFs must be done by centralized management. In some configurations this management software will communicate directly with firmware via side-band interfaces or by wire. In other configurations, management can communicate with the PF software and provide it with information regarding allocated resources. In either case, PF software does not control resources allocated for that PF, but does own full control over distribution of the allocated resource within the PF.

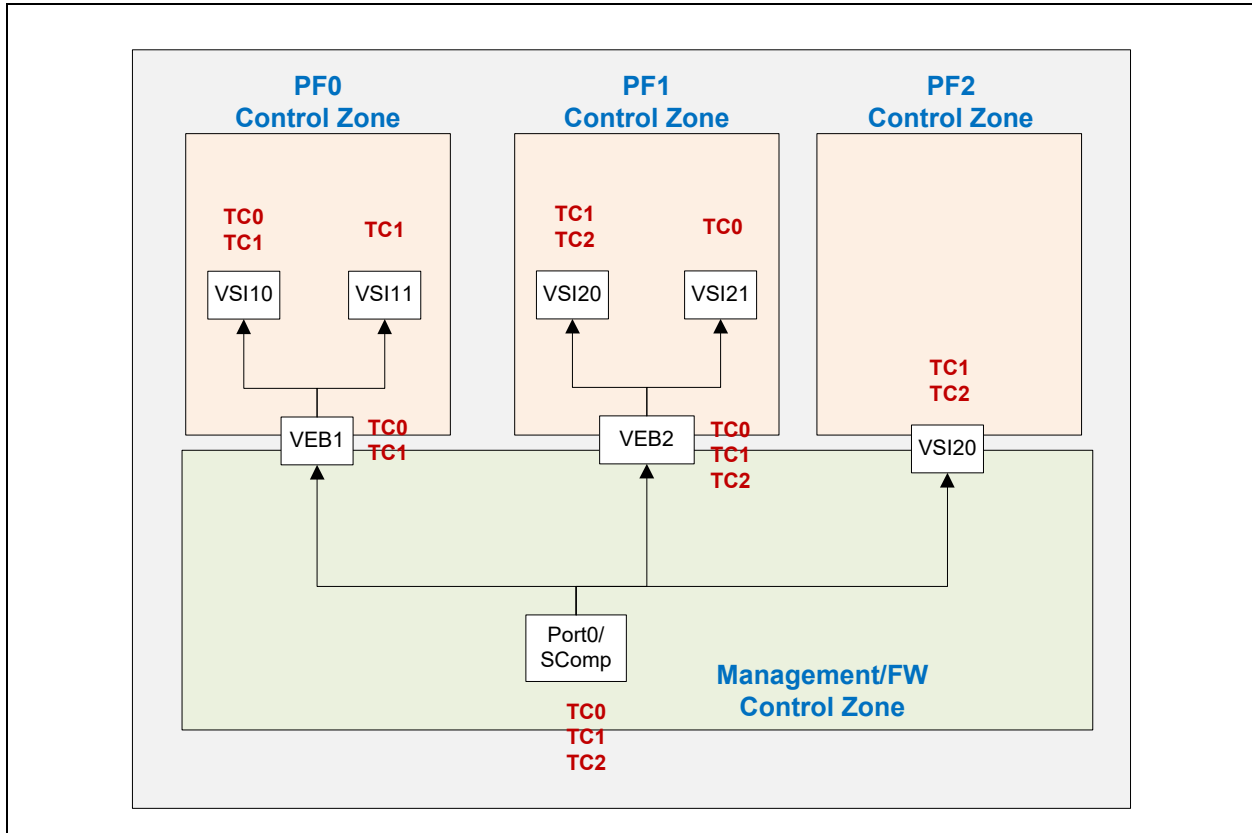


Figure 7-79. Software Control Zone in MFP Mode

Figure 7-79 shows an example of MFP system. Each one of VEBs and VSIs can be configured with relative bandwidth allocation per the enabled TC and its bandwidth limits. Usually this configuration would be done directly by firmware based on instructions from the system management software.

Bandwidth distribution between VSIs allocated within each PF is controlled by the PF driver.

If system management software communicates with the PF driver, and conveys resource allocation for the PF, software can use Admin Queue command described in [Section 7.8.4.14](#) to configure bandwidth allocation for the switching component per enabled TC, or use Admin Queue commands described in [Section 7.8.4.9](#) and [Section 7.8.4.13](#) to enable bandwidth limits. Since the PF driver might not be able to coordinate its configuration with other PFs, it is allowed to provide an absolute bandwidth allocation credit, and to have firmware translate those to the relative bandwidth credits used to program hardware configuration tables.

7.8.5.6 Enabling/Changing Port DCB Configuration

ETS configuration of the physical port can be enabled or modified at any point. This is not expected to be a frequent event, and therefore a transition period to adjust the scheduler configuration to reflect the change is acceptable.

At least one TC must be enabled. If DCB is disabled for the Port, TC0 should be used.

If the port extender (S-comp) is enabled for the physical port, all attributes configured for the port automatically apply for the S-comp (such as ETS configuration, etc.)

In the default flow, firmware negotiates ETS configuration of the physical port, and configures TCs enabled by ETS for the physical port. Default VSI, previously created for the port or physical functions associated with the port, remains associated with TC0 only. Software should use the Admin Queue command described in [Section 7.8.4.8](#) to modify TCs enabled for VSI and a relative bandwidth allocation of VSI within each enabled TCs.

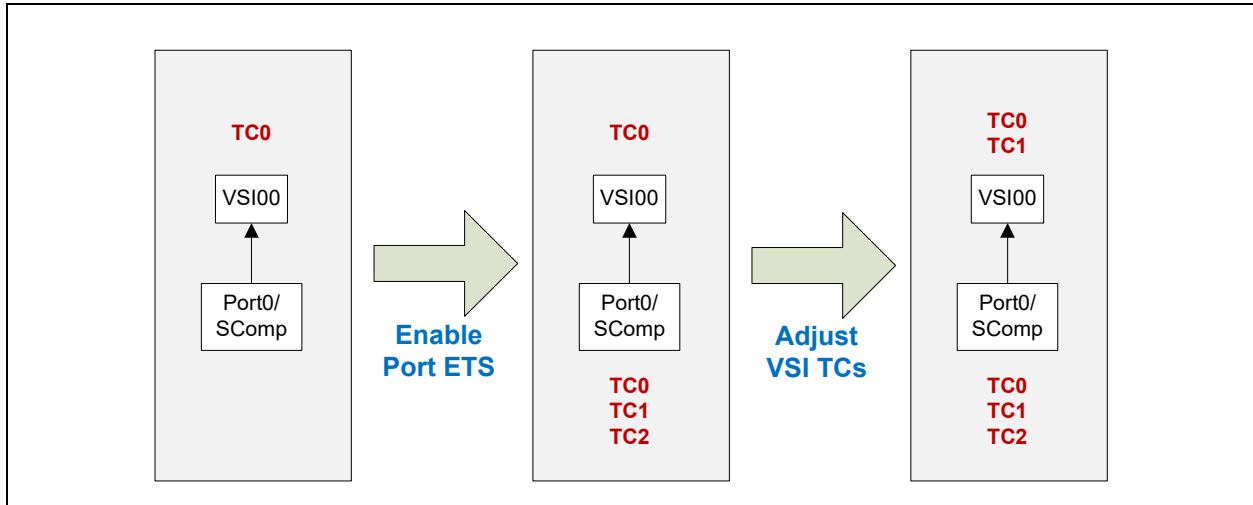


Figure 7-80. Enable Physical Port ETS

If software instantiated additional VSIs or switching components prior to ETS is enabled for the port, or if ETS configuration has been changed, firmware is responsible for updating the ETS configuration of the physical port, and for adding or removing TCs. All previously allocated VSIs and switching components remain associated with previously configured TCs. Software should use the Admin Queue commands described in [Section 7.8.4.8](#) and [Section 7.8.4.14](#) to adjust TCs enabled for VSIs and switching components, and to modify bandwidth allocation per traffic type as required.

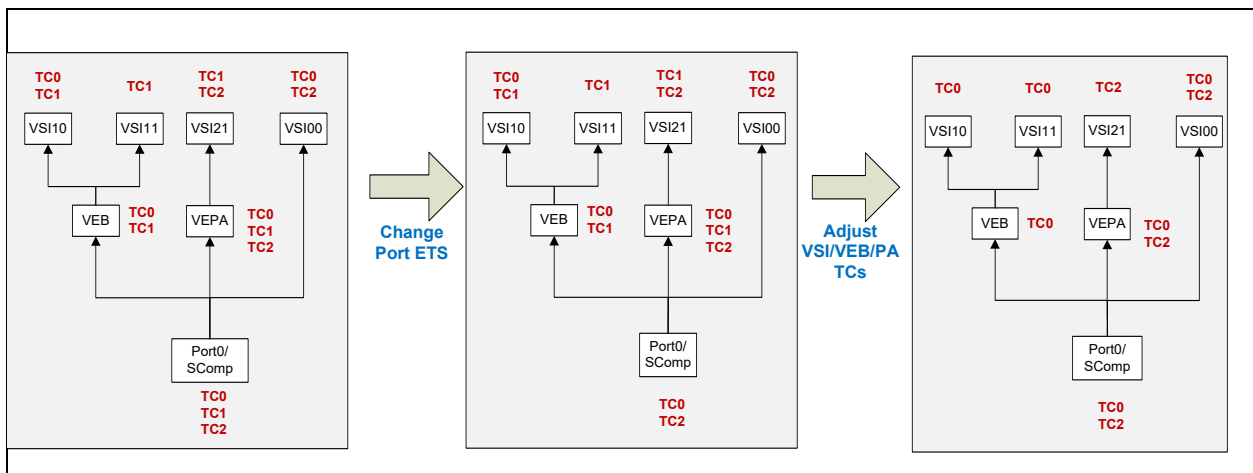


Figure 7-81. Change Physical Port ETS



Scheduler starts scheduling requests for the new TCs added to the Physical Port configuration only after this TC is enabled for one or more VSIs. Adding more TCs to VSI will lead to allocation of new queue sets, which in turn must be associated with transmit queues.

After a TC is removed from the physical port configuration, the transmit scheduler suspends scheduling traffic for all queue sets associated with that TC and then notifies software with an LLDP MIB Change Event. To resume scheduling traffic on transmit queues associated with such queue sets, software will need to adjust TC configuration of VSIs and switching components, and reassign affected transmit queues to queue sets associated with other TCs.

Port DCB configuration can be directly changed by firmware as a result of the DCBX exchange, assuming that a DCBX agent is running in firmware (which is the default configuration). Or it can be configured by software using the Admin Queue command described in [Section 7.8.4.10](#) if software took over the duty of running DCBX agent. Transmit scheduler configuration does not depend on the location of the DCBX agent.

ETS reconfiguration of VSI might move TX queues between TCs. Running this configuration flow while TX pipe is loaded with TX packets belong to this queue (either in the TCB or in the TPB) might cause out of order completion for this TX queue.

Preventing this error requires verification that the TX pipe is drained from any TX packet belonging to re-configured queue or queue set. There are two trivial ways software can verify this draining.

1. The PF can stop feeding the TX ring and wait till all pending work is completed.
2. The PF can disable the TX queue. This provides faster draining mechanism but flushes all pending work.

EMP firmware provides a service which suspends a port's queue sets and drains the TX pipe immediately. This port draining service is used as part of DCBX change (see [Section 7.8.4.11](#) for details).

The general flow of DCBX exchange works this way: If the ETS setting of VSIs is to be changed, the EMP will first drain the TX pipe of the port (initiated by LLDP owner, done using the EMP and TX scheduler). After the port is drained, firmware will notify the PF, which will make those required ETS changes (via the TX scheduler AQC). After the VSI ETS setting is adjusted, the PF will resume its queue sets using AQC, as described in [Section 7.8.4.7](#) and in [Section 7.8.4.8](#).

Some other DCB parameters require Tx pipe draining before re-configuration. DCBX owner (EMP firmware or software) will use the Tx scheduler draining service.

Some DCBX flows require change of queue context configuration. In those cases, software must first disable the queue.

7.8.5.6.1 Tx Scheduler DCB Control Flows

7.8.5.6.1.1 General

DCBX and TX scheduler control flows are declared under some assumptions and decisions as followed:

- For a DCBX exchange event, software runs the similar flow under SFP or MFP.
- Usually, DCBX agent runs in the EMP. Under SFP mode, software might take LLDP agent ownership.
- In cases where software does not take LLDP ownership, the difference between DCBX software flows when running under SFP or MFP should be minimized.
- DCBX events might occur during first connectivity to the network as part of power up flow, link down or link up event, or rarely, the switch deciding to change DCB settings.
- Three typical boot sequences are identified. (The DCBX runs at a different stage in each case):



Case I - Typical MFP power on sequence

EMP sets the initial scheduler configuration - One initial VSI per PF. Detailed flow in [Section 7.8.5.6.1.2](#)

DCBX runs immediately when EMP initializes (when link goes up). Detailed flow in [Section 7.8.5.6.1.3](#)

MFP configuration runs

After Alternate RAM is done a global reset is triggered and EMP reloads configuration

EMP sets the initial scheduler configuration including MFP initial configuration. Detailed flow in [Section 7.8.5.6.1.2](#)

Software boot. Detailed flow in [Section 7.8.5.6.1.4](#)

Case II - Typical SFP boot sequence or MFP system reboot after MFP settings already done

EMP sets the initial scheduler configuration - One initial VSI per PF. Detailed flow in [Section 7.8.5.6.1.2](#)

DCBX runs. Detailed flow in [Section 7.8.5.6.1.3](#)

Software boot. Detailed flow in [Section 7.8.5.6.1.4](#)

Case III - Late DCBX, DCBX parameters changed by the peer, or link event.

EMP sets the initial scheduler configuration - One initial VSI per PF. Detailed flow in [Section 7.8.5.6.1.2](#)

Software boot. Detailed flow in [Section 7.8.5.6.1.4](#)

CBX runs or makes changes in DCBX setting. Detailed flow in [Section 7.8.5.6.1.3](#)

- When the link goes up either during power up flow or while system is already running, all DCBX MIBs are reset to their default values (All TC are in DROP policy, all UPs are mapped to TC#0 and TC#0 is the only active TC for this port). After link up, the DCBX flow runs twice, a) Reset MIBs to default. b) New DCBX exchange with the new link partner.

7.8.5.6.1.2 Reset/Init Flow

- EMP initializes switch and Tx scheduler tables.
- For SFP, EMP builds one initial VSI per active port.
- If MFP mode and Alternate RAM already loaded:
 - EMP builds one S-comp per port and one initial VSI per active PF
 - EMP reads minimum and maximum bandwidth configuration of each PF from the Alternate Ram and sets this in the TX scheduler.
- EMP exposes the queue sets for pre-boot Tx (without active Admin Queue).

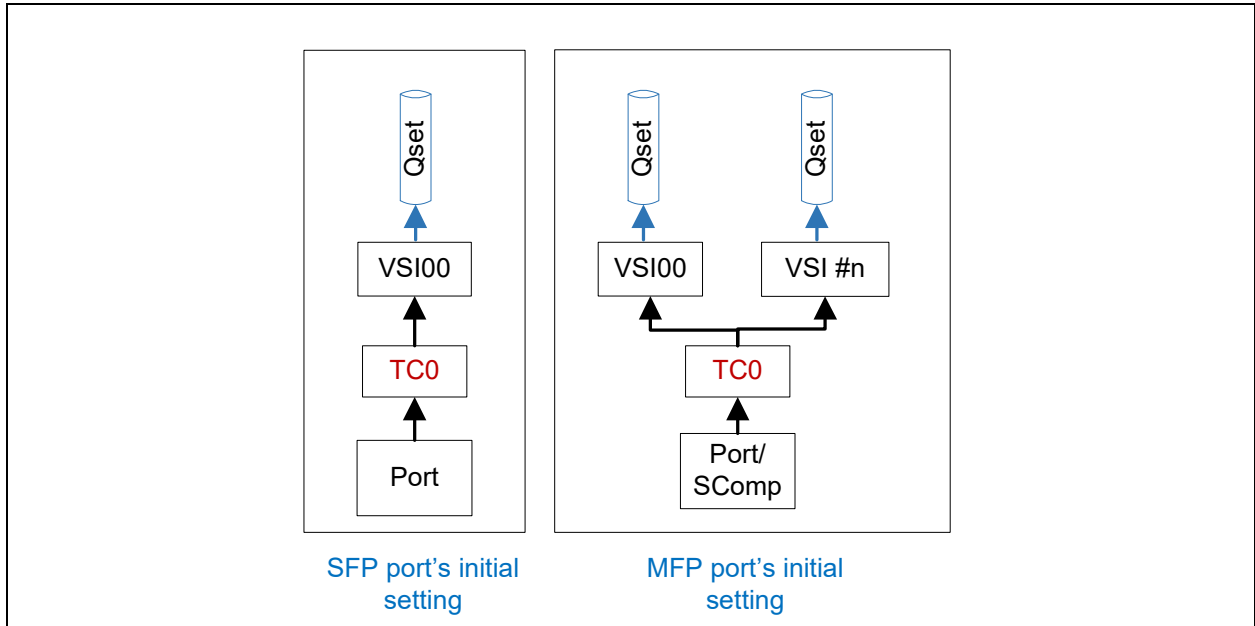
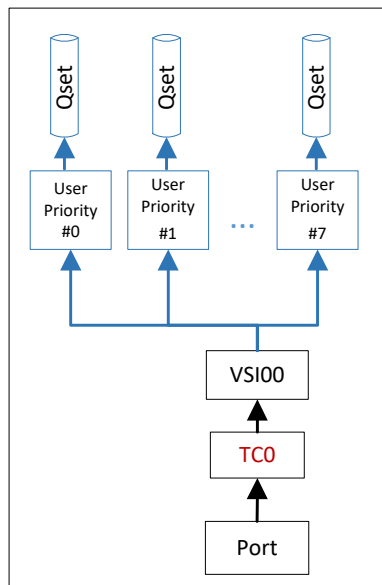


Figure 7-82. Initial Setting in SFP (One Initial VSI/Port) and MFP (One Initial VSI/PF)





7.8.5.6.1.3 Initial Setting in SFP DCBX Exchange

- When EMP owns LLDP:
 - IF port draining is required (due to a change in drop policy, application TLV, UP to TC mapping or port TC mapping), the following occurs:
 - EMP suspends all port TX (all TCs - no credits)
 - Port extender
 - EMP drains port's TX pipeline (wait till all pipe monitor counters are zeroed).
 - EMP re-configures needed DCB settings. (See [Section 7.7.3.3.1](#) step 2 for details).
 - IF port is suspended, EMP resumes manageability Tx traffic.
 - EMP re-configures TX scheduler.
 - Modify Physical Port ETS setting (Enable seepage = True, DCBX TC BW configuration).
 - EMP sends notification of DCBX event to all PFs registered for LLDP MIB notification.
 - This is done via posting LLDP MIB Change Event, after waking up the host if it was not in DO state. (See LLDP MIB Change Event notification details in [Section 7.12.5.2.3.3](#)).
 - LLDP MIB Change Event notification includes a miscellaneous field which marks the PF if the MIB change involved port draining and if per TC pipe flushing was used.
 - Any PF that is not registered for LLDP events must periodically check for LLDP MIBs update status.

The following is relevant only when software is up and running (case III):

- When software owns LLDP (applies only in SFP mode):
 - IF port draining is required, the PF calls Suspend Port's TX Traffic AQC (See [Section 7.8.4.11](#)).
 - EMP suspends all port's TX (all TCs - no credits).
 - EMP stops manageability TX traffic.
 - EMP drains port's TX pipeline, waiting until all pipe monitor counters are zeroed.
 - EMP drains port's TX pipeline (wait until all pipe monitor counters are zeroed).
 - The PF re-configures DCB settings as needed. (See [Section 7.7.3.3.1](#) step 2 for details).
 - The PF instructs EMP to configure TX Scheduler with port's TC setting (via Configure Physical Port ETSAQC, Enable seepage = True).
- The PF computes the needed tree topology changes according to the new UP to TC mapping and PF's UP settings.
- The PF marked as iSCSI service (defined in Alternate Ram: PF Protocol (Offset 0xB)), needs to be opened for both iSCSI TC and LAN TC.
- The PF adjusts VEBs and VSIs per TC setting (in MFP, each PF does this for its resources).
 - Done using Configure VSI Bandwidth Limit per Traffic Type, Configure VSI Bandwidth Allocation per Traffic Type, Configure Switching Element Bandwidth Limit per Traffic Type or Configure Switching Element Bandwidth Allocation per Traffic Type commands.
 - As stated in the commands: Configure VSI Bandwidth Limit per Traffic Type, Configure VSI Bandwidth Allocation per Traffic Type, software should not request changing the TC configuration of VSI leading to release or remapping of TCs, prior to disassociating queues with affected queue sets with one exception: when one and only one TC node is moved.
 - Upon disabling the queues, the PF should not wait for completion because the queue disable flow is not completed by the hardware until after the TX activity is resumed at the end of the DCB reconfiguration flow.
 - In case the commands Configure VSI Bandwidth Limit per Traffic Type or Configure VSI Bandwidth Allocation per Traffic Type are called and remap only one TC node, EMP will give this TC node its original queue set handle. This allows the PF to skip the step of disabling and enabling all queues.



- The commands Configure VSI Bandwidth Limit per Traffic Type and Configure VSI Bandwidth Allocation per Traffic Type can be used to expand a VSI's TC setting. EMP returns the queue set handle for each of the enabled TCs, including the new used TCs.
- The PF copies the new queue set handles to the contexts of the TX queues belonging to the new established queue sets. When a queue context needs to change its queue set (moving between queue sets), software must disable, reconfigure and re-enable the queue.
- Please see [Figure 7-79](#) for an explanation of control zones in MFP. Under MFP, in a PF's TC setting, the minimum BW and maximum bandwidth configuration values are derived from Alternate Ram MFP configuration data. When the PF tries to configure the TC or bandwidth configuration of an entity in firmware control zone (VEB or VSI connected to the port extender), EMP firmware ignores the parameters provided in the command. Instead, EMP computes the TC setting and bandwidth configuration from the Alternate Ram function's configuration. The EMP returns the actual TC and bandwidth setting. The same function call is used for both SFP and MFP, although in MFP, the parameters are not used. This is done for compatibility purposes, and provides an identical flow in software under both SFP and MFP).
- After the configuration completes, the PF resumes its suspended Tx traffic using the Resume PF Traffic Admin Queue command. (See [Section 7.8.4.12](#)).
 - If the PF was required to disable queues during the configuration flow, then wait for the queue disable flow to be completed by hardware.
 - Enable the active queues.



7.8.5.6.1.4 Software Boot

- The PF reads the DCBX MIBs, including UP-TC mapping.
 - To get consistent MIB data, the PF must verify that LLDP owner is not currently processing LLDP message by polling PRTDCB_GENS.DCBX_STATUS until it is set to done.
- Before making any change to VSI settings, the PF reads the initial configuration and configures all queue sets.
 - Tree topology reading is done using the Get Switch Configuration command (see [Section 7.4.9.5.3.1](#)).
 - For each VSI, the PF must call Query VSI Bandwidth Configuration (see [Section 7.8.4.15](#)).
 - The response buffer includes:
 - A bitmap of which TCs are enabled in this VSI.
 - A queue set handle for each enabled TC.
 - A bitmap of which Enabled TCs are suspended due to a DCBX event.
 - The PF must verify that a queue set is established for each enabled TC.
 - For each enabled TC, the PF must copy the queue set handle value to all queues belonging to the queue set (before enabling the queue).
- The PF determines whether to move/add any VSI to a different TC. This is done using the Configure VSI Bandwidth Limit per Traffic Type, Configure VSI Bandwidth Allocation per Traffic Type commands.
 - In case the commands Configure VSI Bandwidth Limit per Traffic Type, Configure VSI Bandwidth Allocation per Traffic Type are called and move only one TC node, EMP will give this TC node its original queue set handle. This allows the PF to skip the step of disabling and enabling all queues.
 - The PF resumes its suspended VSIs using AQC Resume PF traffic. (See [Section 7.8.4.12](#)).
- Whenever the PF needs to change the TC setting of a VSI (not as a response to DCBX event, This VSI is NOT suspended), the PF MUST drain all TX queues belonging to moved TCs (in the VSI) before changing its TC settings. (Changing any setting of a queue set while it has packets in the TX pipe might cause out of order completion).
- Any topology change is done using the Add VSI or Add VEB command (for the Enabled TCs field) that defines the TCs for the VSI or VEB.
 - EMP adds VEB, VSI, queues accordingly.
 - As a response to ADD VSI command, EMP provides the queue set handles of the enabled TCs.
 - For each enabled TC, the PF must copy the queue set handle value to all queues belonging to the queue set.

7.8.5.7 Transmit Scheduler Resource Allocation Control

Each PF is configured with a limited number of resources. The The X710-TM4 scheduler allows flexible resource allocation, but due to limited number of queue sets supported, it allows the allocation of an average of two queue sets per VSI.

The number of VSIs dedicated per PF is a Switch Configuration parameter configured via NVRAM. The rest of the available VSIs are shared on the First-Come-First-Serve bases.

Scheduler configuration firmware only controls resources allocated for PF, and allows the PF driver to distribute those resources within the PF.



The allocation of queue sets per PF is proportional to the allocation of VSIs, assuming average of two queue sets per VSI. Shared queue sets can be allocated to PFs on First-Come-First-Served bases, similar to VSIs.

7.8.5.8 Scheduler Configuration Schemes

The ETS-based scheme allows ETS configuration for the switching component or VSI directly connected to the physical port, and bandwidth allocation of other VSIs and switching components within each traffic type that can be either UP or TC enabled for that VSI or switching component. For details, see [Section 7.8.2.1](#).

Software should not attempt to configure bandwidth management attributes that are not supported by the configured scheme. Firmware will reject any attempts to perform invalid operations, and will return an EPERM error described in [Section 7.8.4.4](#).

7.8.5.8.1 ETS-Based Scheme - Relative Bandwidth Credits Calculation

ETS-based configuration allows X710-TM4 software to configure relative bandwidth allocation for each switching component, and VSI within each traffic type (UP or TC). Software should use Admin Queue commands described in [Section 7.8.4.8](#) and [Section 7.8.4.14](#).

X710-TM4 software should be able to configure ETS-based transmit scheduler based on either:

- An absolute bandwidth allocation (in GB/s or Mb/s) for VSIs and switching components within each TC or UP enabled for the respective VSI or switching component.
- An absolute bandwidth allocation for VSIs and switching components, and ETS configuration for each VSI and switching component.

In either case, any configuration provided by system management software will need to be translated to relative bandwidth allocation of VSIs and switching components within TCs or UPs, and expressed in relative bandwidth allocation credits.

[Figure 7-83](#) shows an example of an ETS-based scheme bandwidth configuration and the process of relative bandwidth credits calculation.

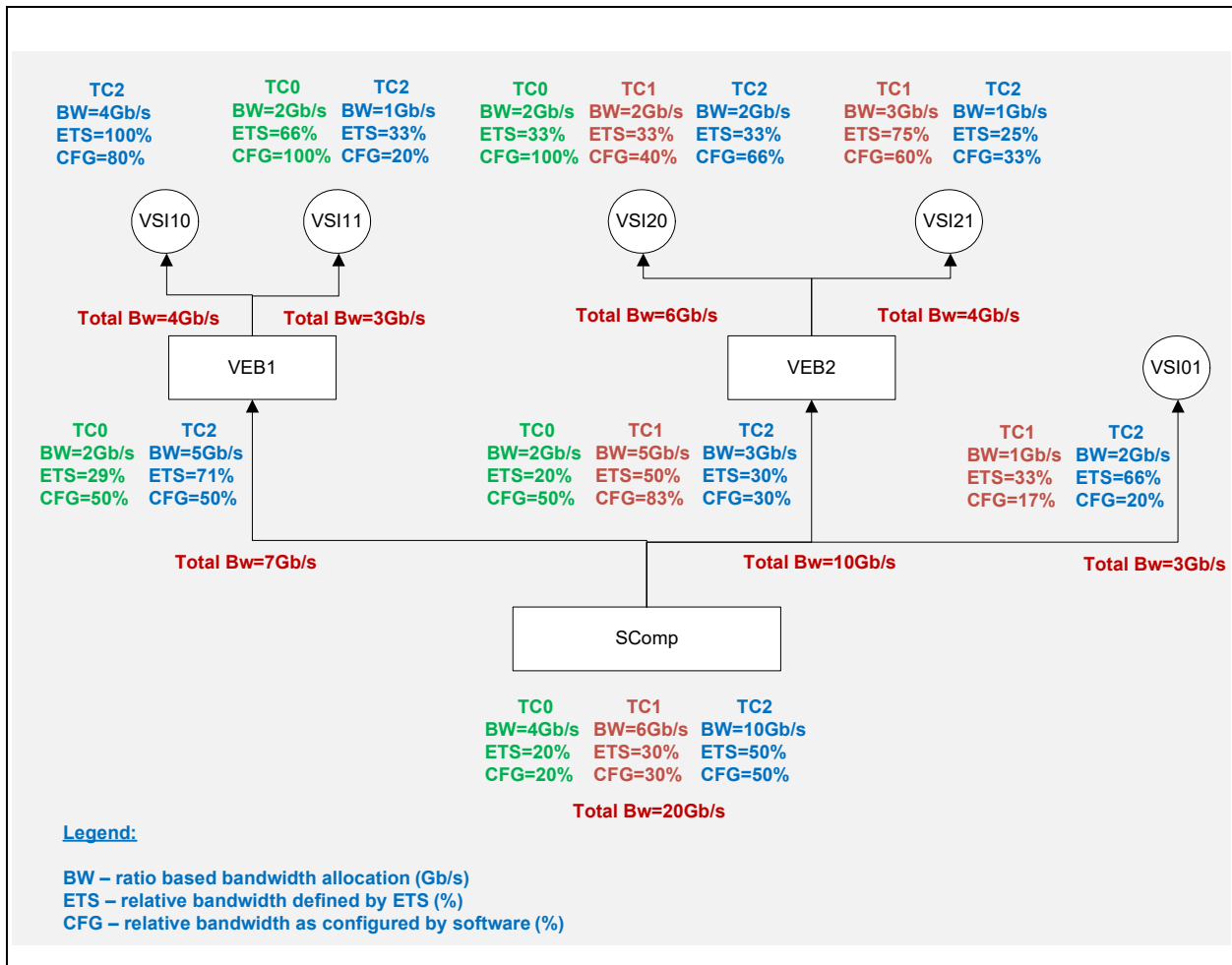


Figure 7-83. ETS-Based Scheme (Relative Bandwidth Calculation Example)

Figure 7-83 shows a single port configuration with S-comp. Two VEB switching components are connected to S-comp (VEB1 and VEB2), along with VSI (VSI01) directly attached to S-comp. Each VEB has two VSIs (VSI10 and VSI11, and VSI20 and VSI21 respectively).

This diagram for each switching component and VSI shows a per-TC bandwidth allocation using 3 terms:

- ETS - relative bandwidth allocation per TC within VSI/switching component in percentage with respect to other TCs. This can be provided by the system software along with total bandwidth allocated for the switching component and VSI.
- Bandwidth - bandwidth allocation per TC in Gb/s, based on the physical wire speed of 20 Gb/s, and hierarchical bandwidth distribution specified by ETS. This can be provided by system management software, or can be calculated based on the total bandwidth allocation for either the switching component, or the VSI and ETS configuration.
- CFG - calculated hierarchical relative bandwidth allocation of VSIs and switching components within each TC, in percentage with respect to other VSIs and switching components on the same hierarchy level.



- Total Bandwidth - provided by system management software along with ETS configuration for the switching components and VSIs.

Note: ETS-Based configuration does not guarantee bandwidth allocation for VSIs and switching components in ETS-Based configuration, but it can use bandwidth allocation provided by system software to configure VSI and switching components relative bandwidth allocation within each TC or UP.

If the physical port is configured to two level ETS (sothat bandwidth is allocated for TCs, and then distributed between UPs within same TC), then the VSI and switching component bandwidth is allocated per UP.

If the physical port is configured to single level ETS (with bandwidth distribution between TCs only), X710-TM4 software can assume a logical one-to-one mapping of UPs to TCs for the bandwidth distribution purpose, and can use Admin Queue commands described in [Section 7.8.4.8](#) and [Section 7.8.4.14](#) to configure the VSi and switching component bandwidth allocation within a TC.

In either case, relative credits calculation should be done following this algorithm:

- If system software provides absolute bandwidth allocation per TC or UP:
 - The X710-TM4 software should calculate a relative bandwidth allocation with respect to the parent switching entity within each traffic type (TC or UP).
 - Translate relative bandwidth allocation to credits, minimizing number of credits allocated.
- If the system software provides an absolute bandwidth allocation for VSI or switching component and ETS:
 - X710-TM4 software should calculate an absolute bandwidth allocation per traffic type within VSI or switching component. This would be the same as an absolute bandwidth allocation of VSI or switching component within traffic type (TC or UP). Then it should calculate a relative bandwidth allocation with respect to the parent switching entity within each traffic type (TC or UP).
 - Translate relative bandwidth allocation to credits, minimizing number of credits allocated.

Table 7-182 shows an example of a relative bandwidth credits calculation for the ETS-Based system configuration shown on [Figure 7-83](#). This example assumes that the system management software provides a total bandwidth allocated for each VSI and switching component, and for ETS.

Table 7-182. ETS-Based Scheme-Relative Bandwidth Calculation Example

Switching Element Name	Parent Switching Element Name	Total BW	ETS (TCs)	ETS (%)	ETS (Gb/s)	Relative BW within TC	Relative Credits within TC	Comments
SComp	Physical Port	20Gb/s	TC0	20%	4Gb/s	100%	NA	No need to configure relative bandwidth allocation within traffic type for switching element attached directly to physical port
			TC1	30%	6Gb/s	100%	NA	
			TC2	50%	10Gb/s	100%	NA	
VEB1	SComp	7Gb/s	TC0	29%	2Gb/s	50%	1	VEB1 shares TC0 bandwidth with VEB2, 50% each, therefore 1 relative credit will correctly reflect a relative bandwidth allocation between VEB1 and VEB2 (1:1)
			TC2	71%	5Gb/s	50%	5	VEB1 shared TC2 bandwidth with VEB2 and VSI01, with ratio 5:3:2, which is reflected by relative credits



Table 7-182. ETS-Based Scheme-Relative Bandwidth Calculation Example

Switching Element Name	Parent Switching Element Name	Total BW	ETS (TCs)	ETS (%)	ETS (Gb/s)	Relative BW within TC	Relative Credits within TC	Comments
VEB2	SComp	10Gb/s	TC0	20%	2Gb/s	50%	1	VEB2 shares TC1 bandwidth with VSI01, relative ratio is 5:1, which is reflected in relative credits
			TC1	50%	5Gb/s	83%	5	
			TC2	30%	3Gb/s	30%	3	
VSI01	SComp	3Gb/s	TC1	33%	1Gb/s	17%	1	
			TC2	66%	2Gb/s	20%	2	
VSI10	VEB1	4Gb/s	TC2	100%	4Gb/s	80%	4	VSI10 shared TC2 bandwidth allocated for VEB1 with VSI11, with relative ratio 4:1, which is reflected in allocated relative credits
VSI11	VEB1	3Gb/s	TC0	66%	2Gb/s	100%	1	VSI11 consumes all TC0 bandwidth allocated for VEB1, and therefore can be configured with 1 relative credit
			TC2	33%	1Gb/s	20%	1	
VSI20	VEB2	6Gb/s	TC0	33%	2Gb/s	100%	1	VSI20 consumes all TC0 bandwidth allocated for VEB2, and therefore can be configured with 1 relative credit
			TC1	33%	2Gb/s	40%	2	VSI20 shared TC1 bandwidth allocated to VEB2 with VSI21 with relative ratio 2:3, which is reflected in relative credits
			TC2	33%	2Gb/s	66%	2	
VSI21	VEB2	4Gb/s	TC1	75%	3Gb/s	60%	3	VSI21 shared TC2 bandwidth allocated to VEB2 with VSI20 with relative ratio 1:2, which is reflected in relative credits
			TC2	25%	1Gb/s	33%	1	

Table cells with gray background show numbers calculated by X710-TM4 software.



NOTE: *This page intentionally left blank.*



7.9 Host Memory Cache (HMC)

The X710-TM4 uses host memory as backing store for a number of context objects used to track the queue state. The HMC is the component responsible for managing the LAN context objects stored in host memory. The HMC manages host memory on a per PCI function basis and further breaks down each PCI function’s HMC memory space into memory used to manage each context object that is in use for a given PCI function. Host software is responsible for the allocation of the host pages used by the HMC before accessing a specific object. Additionally, the amount of memory that can be used for HMC backing store for a specific function is dictated by the active resource profile, which is in turn determined by the software driver’s operating environment and the number of currently active PCI functions. Resource profiles can be selected at driver initialization time.

7.9.1 Host Memory Usage

The HMC requires backing store for numerous data structures to be resident in host memory to perform its functions. [Table 7-183](#) lists the data structures and the amount of memory that needs to be allocated for each data structure. The HMC PCI Function Type column indicates if the HMC object (and the associated backing store pages) is located only in the PF HMC object space, or if it is located in both the PF and the VF HMC object space. In general, all LAN objects for both PFs and VFs are located in the PF HMC object space. The resources can be sparsely populated. For example, if a function is allotted 512 QPs and only 8 are used, then only 4K of memory needs to be allocated, instead of the entire memory for all 512 QPs. See [Section 7.9](#) for more information on the HMC resource allocation policies for LAN.

Table 7-183. HMC Objects

HMC Object	HMC Object Location	Size (Bytes)	Max Quantity	Description
LAN Transmit Queue	PFs	128	1536 per device	The PF owns the objects for the associated VFs. LAN absolute queue numbers assignment to PFs are determined by the programming of the PFLAN_QALLOC registers. HMC object indexes match the LAN queue indexes. See Section 7.9 for more details on LAN queue allocation.
LAN Receive Queue	PFs	32	1536 per device	The PF owns the objects for the associated VFs. See the previous description of LAN Transmit queues for details on the relationship between absolute LAN QP numbers and HMC object indexes.

To access (and cache in on-chip memory) the data structures defined in [Table 7-183](#), the HMC uses the concept of private memory address space. The X710-TM4 has an 8 GB private memory address space that can be sparsely backed with host memory based on actual context usage. Software device drivers do not need to allocate pages for HMC objects that are not currently being used by the software device driver. The private memory address space is first broken down by PCI function, then by object or data structure type, and finally by object index. The portion of the private memory address space allocated to a particular PCI function is termed Function Private Memory (FPM). FPM objects for VF LAN are located in the associated PFs FPM.



Figure 7-84 shows how the X710-TM4 provides the address mapping between Private Memory and Host Physical Addresses. PM Address on the left side of the figure shows X710-TM4 Private Memory address from 0 to 8 GB-1. The X710-TM4 works with PM address space internally which is converted to Host Physical Addresses in order to access host memory.

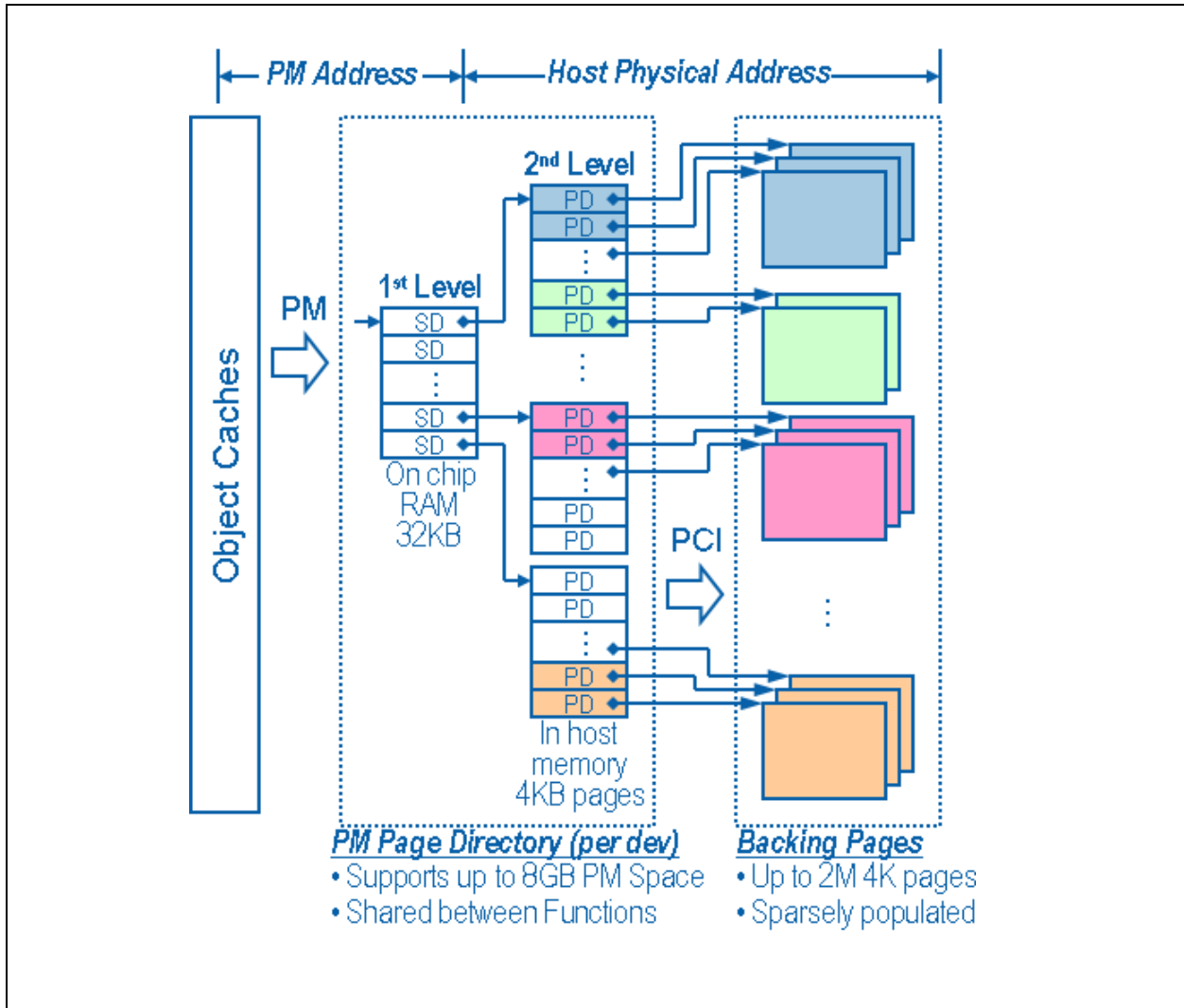


Figure 7-84. Host Memory Cache Private Memory Address Space



The left portion of [Figure 7-84](#) shows portions of the HMC that are resident on-chip. This portion includes the actual object caches that retain portions of the data from host memory to improve performance, and the Segment Descriptors (SD). The SDs reside in a 32 KB RAM on-chip called the Segment Descriptor Table. The Segment Descriptor Table holds 4096 pointers to host memory pages. Unique ranges of sequential SDs in the Segment Descriptor Table are each allocated to an active PCI function. The Segment Descriptor Table is the first level of private memory address translation provided by the X710-TM4. SDs are programmed using the PFHMC_SDCMD, PFHMC_SDDATALOW and PFHMC_SDDATAHIGH registers. Everything to the right of the Segment Descriptor Table in [Figure 7-84](#) resides in host memory. Each PCI function has a set of registers (GLHMC_SDPART[n]) that define the base and number of SDs that belong to the PCI function. The GLHMC_SDPART[n] registers are programmed from NVM. The X710-TM4 provides range checking for each internal access to ensure that a given PCI function is never allowed to access memory outside of its valid range of SDs. The X710-TM4 manages the SD base and number registers internally, based on the resource profile loaded from NVM.

The second level of private memory address translation provided by the X710-TM4 are Page Descriptors (PDs). Each SD points to a single host page that is divided into 512 PDs, which are in turn 64-bit physical memory addresses. Each PD points to a backing page for the private memory address space. The total 8GB private memory address space is derived using a fully populated Segment Descriptor Table pointing to 4096 4 KB Host pages that hold the 2M PDs. Each of the 2M PDs point to host memory backing pages for a total of 8GB of address space. As previously mentioned, there is no requirement to populate all SDs or PDs with memory if the portion of private memory address space is not in use by software. The format of the PD structure in host memory is listed in [Table 7-184](#).

Table 7-184. HMC Page Descriptor Format

Byte Offset	[Bit Range]Field Name
0	[63:12]HMC Backing Page Physical Address [11:1]Reserved [0]PD Valid

The HMC Backing page physical address is the address of a driver allocated page that holds HMC object context. This address must be aligned to a 4KB address in host memory. The PD Valid bit allows software to sparsely populate the PD entries on an as needed basis once HMC context objects are needed. Software must allocate host memory pages which hold packed arrays of PDs as shown in [Figure 7-84](#). The physical addresses of these PD pages are used to populate SD entries by using the PFHMC_SDCMD, PFHMC_SDDATALOW and PFHMC_SDDATAHIGH registers. For more information on how to program SDs using these registers, see [Section 7.9.3](#).

The private memory is further divided into separate PCI Function Private Memory (FPM) Addresses. A PCI function can be either physical function or virtual function. The first 16 FPM address spaces are reserved for PFs. NIC VF HMC objects are located in the PF FPM.

[Figure 7-85](#) shows how the private memory address space is divided up for each PCI function. The smallest amount of private memory that can be allocated to a function is 2 MB (1 SD). The maximum that could be allocated to a function would be the entire segment table, in which case no other function would have any private memory resources. Note that the object caches address the HMC objects, using HMC function number to determine the correct FPM. The FPM identifies the range of private memory address space that belongs to a PCI function. Since each SD represents 2 MB of HMC PM address space, the FPM also identifies the range of SDs that belong to a PCI function.

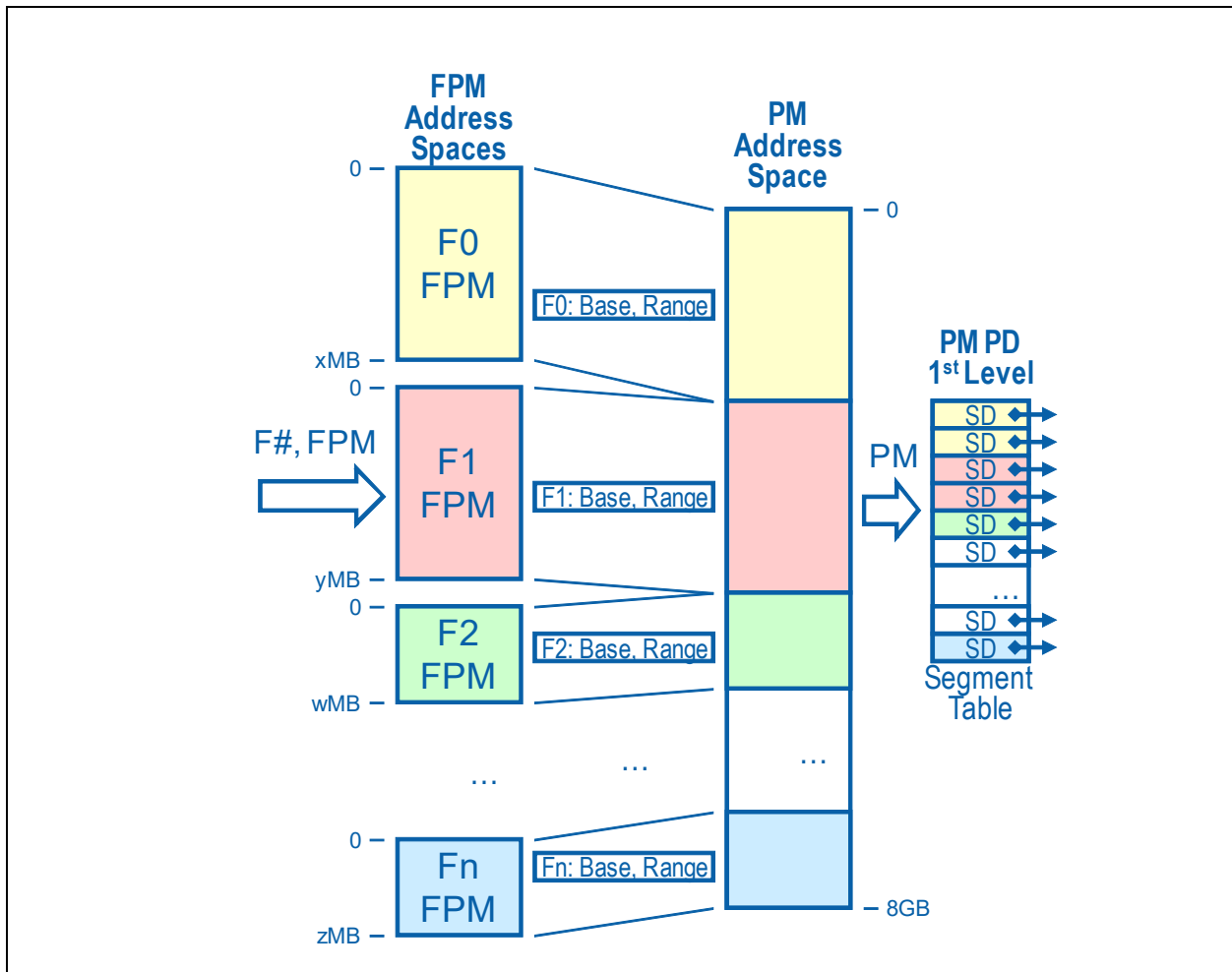
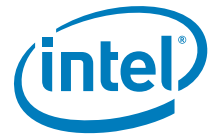


Figure 7-85. Host Memory Cache Function Private Memory Space

Each PCI Function private memory space is further divided into separate memory spaces for each object in host memory. Each PCI function has a set of registers per function that define the object’s base address in FPM space and the bounds (or maximum number of entries) of a particular object. [Figure 7-86](#) depicts some of the current objects that reside in the private memory space (See [Table 7-183](#) for a complete list of the objects). The FPM address is calculated based on the object type (which identifies the object base register) and object index (the FPM base has already been calculated). Ultimately, the FPM base address, object base address, object size, and object index are all used to determine the private memory address.

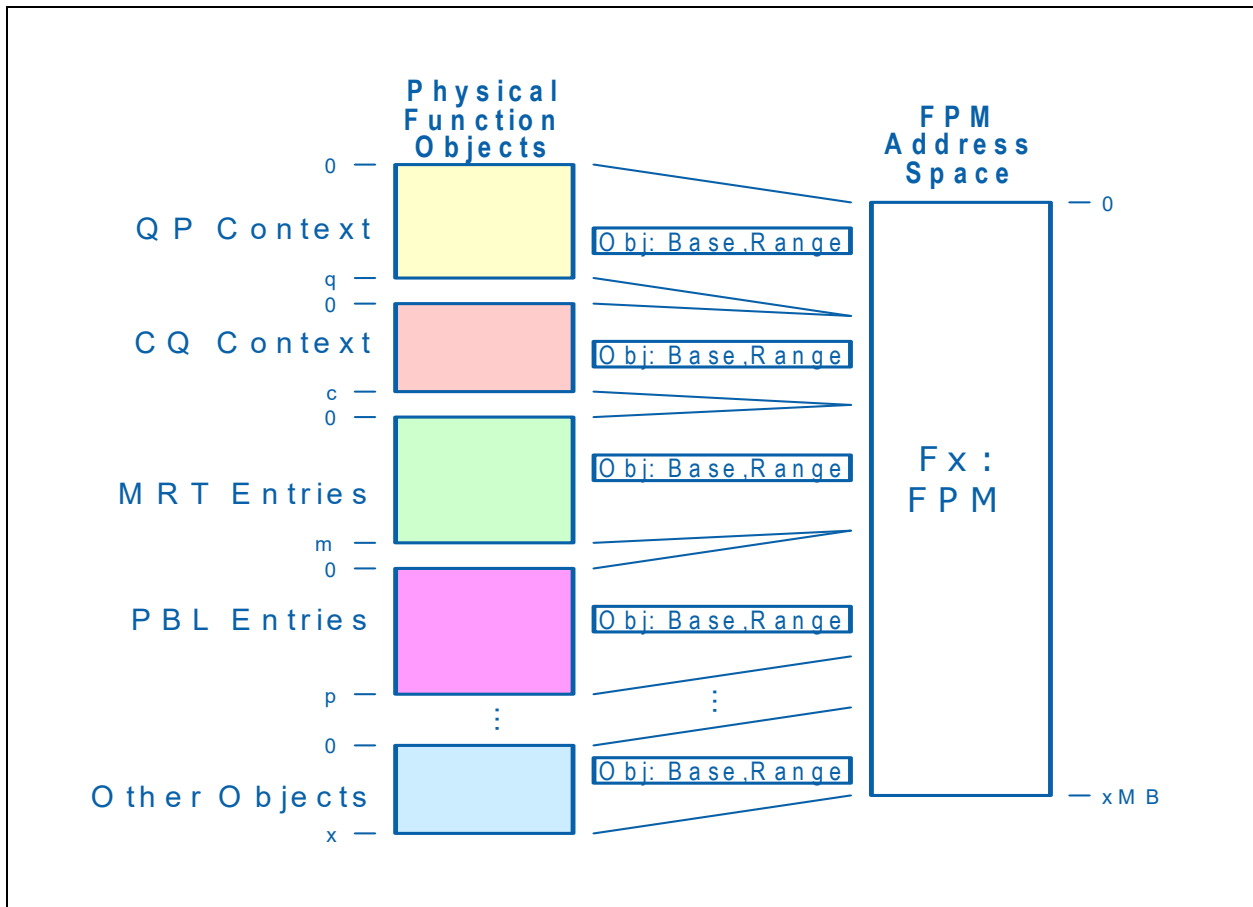


Figure 7-86. Host Memory Cache FPM Object Access

Figure 7-87 shows the decoding of the Private Memory Address into Host Address. Additionally, Figure 7-87 shows an SD addressing a private memory space backing page directly instead of using the second level of indirect addressing (PD). Each PCI function can set any SD within its range of SDs to point to a PD or directly point to a backing page. The segment type is specified in the PFHMC_SDDATALOW.PMSDTYPE register field. To reduce overhead incurred while accessing HMC objects, the direct segment approach can be used for PCI functions that do not have large FPM space requirements. Additional usage of the direct segment approach is possible if the driver is able to allocate a physically contiguous range of pages large enough to hold the entire PD space needed to support the FPM required by the driver loading on a specific PCI function. This mode prevents an additional address lookup, and to increases the performance if the driver happens to allocate a block of physically contiguous memory or the if the operating system has support for 2 MB pages.

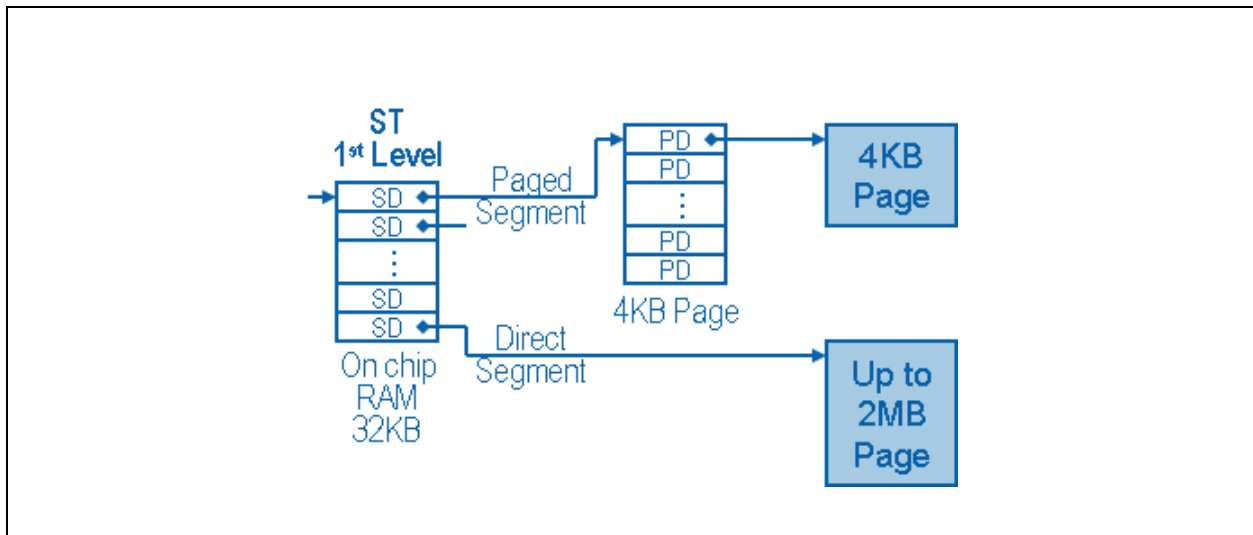


Figure 7-87. Host Memory Cache Direct Segment

See Section 7.9.3 for more details on the specific formats of the SD entries for paged and direct addressing modes.

7.9.2 Function Private Memory Space Configuration

As soon as NVM has set the default profile, the driver can then continue with the second step of HMC configuration, breaking down the FPM space into individual object regions. To do this, the driver must perform the following steps:

1. Determine the HMC function index to be configured. In the case of a PF, the HMC function number is equal to the PCI function number.
2. For LAN queue objects, read the PFLAN_QALLOC register associated with the PF to find the base queue index and number of queues associated with the PF.
3. Each FPM object size register is written with the minimum of the values determined in steps 2-4, or the values the actual driver needs.
4. Write the base and count registers for each LAN object based on the maximum object index that would be used for the PCI function (HMC PM LAN objects are indexed with the absolute queue number).
5. Program LAN register.

Table 7-185 lists all the HMC objects and the registers used to determine the object location within FPM space, and the size and limit of each object.



Table 7-185. FPM Object Registers

HMC Object	Base Register Array	Object Count Register Array	Maximum Object Count Register	Object Element Size Register
LAN Transmit Queue	GLHMC_LANTXBASE	GLHMC_LANTXCNT	PFLAN_QALLOC	GLHMC_LANTXOBSZ
LAN Receive Queue	GLHMC_LANRXBASE	GLHMC_LANRXCNT	PFLAN_QALLOC	GLHMC_LANRXOBSZ

7.9.2.1 Programming the HMC FPM Base Registers

Host software is responsible for setting up the GLHMC_{object}CNT and GLHMC_{object}BASE registers for LAN objects. All settings of FPM registers impact only the function associated with the registers. In other words, the driver on a given PCI function must only program the GLHMC_{object}CNT and GLHMC_{object}BASE registers for HMC PMs that are owned by that same PCI function. The FPM base of the first HMC object (GLHMC_LANTXBASE) for each PCI function is always 0. The FPM base of subsequent HMC objects increment from previous HMC object base, the number of elements for the previous HMC object, and the size of the previous HMC object element. Additional rounding is necessary to get to the next FPM address that is properly aligned for the HMC object under consideration. Table 7-186 lists the FPM object order that must be maintained for proper HMC operation, and also the alignment requirements for each object.

Table 7-186. FPM Object Order and Alignment

HMC Object Order	HMC Object	Alignment Requirement
0	LAN Transmit Queue	512B
1	LAN Receive Queue	512B
2	Reserved	512B
3	Reserved	512B

As a partial example of how the X710-TM4 must program the registers listed in Table 7-186 the following steps would be taken to program the first three HMC objects:

1. Program GLHMC_LANTXBASE = 0
2. Program GLHMC_LANRXBASE = $\text{ROUNDUP}_{512}((\text{GLHMC_LANTXBASE} * 512) + (\text{GLHMC_LANTXCNT} * 2^{\text{GLHMC_LANTXOBSZ}})) / 512$

Note: The software device driver can choose to set the GLHMC_{obj}CNT register to 0 if it does not need to utilize an object. For example, only the GLHMC_LANTXOBSZ and GLHMC_LANRXOBSZ register need be non-zero if only LAN resources are needed.

7.9.3 Populating HMC Backing Pages

As soon as the function private memory space has been programmed (Section 7.9.2), the software device driver must populate the HMC backing pages for the PCI function that it is initializing. The first step in this phase of initialization is to allocate 4 KB pages for the PDs. Each 4 KB PD page holds 512 PDs and occupies a single SD entry. After a 4 KB page has been allocated, initialized to zero and pinned by software, then the PFHMC_SDCMD, PFHMC_SDDATALOW, and PFHMC_SDDATAHIGH registers are



used to populate the SD table for the PCI Function. A software device driver on a given PCI function must only manipulate SD table entries that are allocated for that PCI function via the SD partitioning process. SDs are addressed on a per PCI function basis starting and 0 and is limited by PMSD_MAX. In other words, software is not aware of which portion of the SD table it is using. Accesses outside the SD range configured by NVM using the PFHMC_SDCMD registers are ignored (operation will not be performed) by the X710-TM4 and an error is returned in the completion for the operation that is in error. The second step in this phase of driver initialization is to allocate additional host pages for backing HMC FPM objects for use by the X710-TM4 before the driver attempts to access the object. The breakdown of the FPM address into components is listed in Table 7-186.

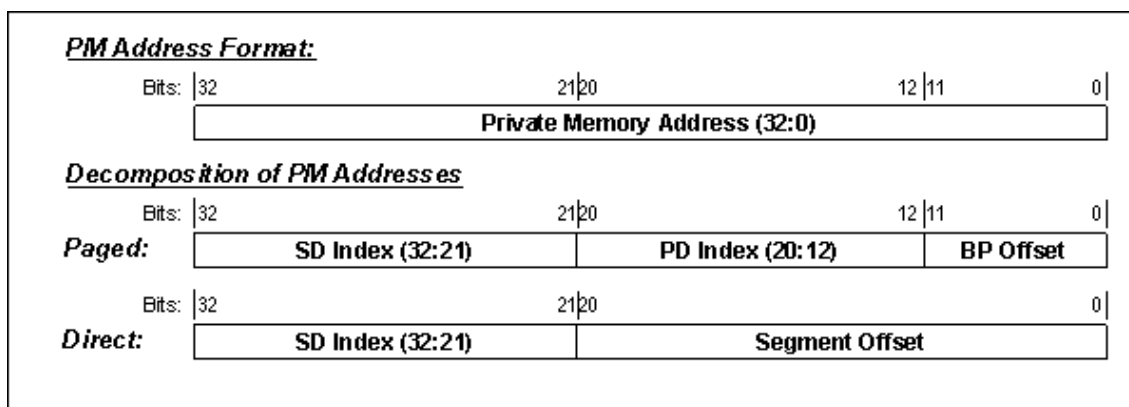


Figure 7-88. FPM Address Decomposition

The identification of which SDs to populate and which HMC FPM backing pages to populate in the PD pages can be calculated as follows in the paged scenario:

$$\text{FPM_object_address} = (\text{GLHMC_}\{\text{object}\}\text{BASE} * 512) + (2^{\text{GLHMC_}\{\text{object}\}\text{OBSZ}} * \text{element_index})$$

$$\text{SD_index} = \text{INT}(\text{FPM_object_address} / 2\text{MB})$$

$$\text{PD_index} = \text{INT}(\text{FPM_object_address} / 4\text{KB}) \& 0\text{x}1\text{FF}$$

HMC_PM_index = PF index or the HMC VF FPM index

An example of populating backing pages in the HMC is shown assuming that a driver wants to allocate FPM backing pages for 512 LAN Receive Queues starting at index 0:

1. Allocate one PD page (capable of holding 512 backing pages of 4KB each)
2. Identify the first SD necessary
 - a. $\text{FPM_object_address} = (\text{GLHMC_LANTXBASE}[\text{HMC_PM_index}] * 512) + (2^{\text{GLHMC_LANTXOBSZ}} * \text{GLHMC_LANTXCNT}[\text{HMC_PM_index}])$
 - b. $\text{FPM_object_limit} = \text{FPM_object_address} + (2^{\text{GLHMC_LANTXOBSZ}} * 512)$
 - c. $\text{SD_index} = \text{FPM_object_address} / 2\text{MB}$
 - d. $\text{Last_SD_index} = ((\text{FPM_object_limit} - 1) / 2\text{MB})$
3. Allocate, zero and pin a host memory page (PD page) for each SD needed from SD_index to Last_SD_index
4. Calculate the number of PDs that need to be allocated
 - a. $\text{FPM_PD_index} = (\text{FPM_object_address} / 4\text{KB}) \& 0\text{x}1\text{FF}$



- b. $FPM_PD_limit_index = ((FPM_object_limit - 1) / 4KB) \& 0x1FF$
- c. $FPM_PD_count = FPM_PD_Limit + 1 - FPM_PD_index$
5. Initialize the PDs
 - a. Allocate/zero/pin FPM_PD_count pages (these are the FPM object backing pages).
 - b. Initialize each of the PDs with the physical address of a page allocated in step 5a and set the PD valid bit (see [Table 7-184](#) for the format). The PDs are in the PD pages allocated in step 3.
6. Update the SD table using the PFHMC_SDCMD, PFHMC_SDDATALOW, and PFHMC_SDDATAHIGH registers for each PD page allocated in step 3.
 - a. Write the most significant 32 bits of the physical address of the PD page to the PFHMC_SDDATAHIGH
 - b. Write the last significant 32 bits of the physical address of the PD page to the PFHMC_SDDATALOW ensuring that the lower 12 bits are 0.
 - c. Write 512 to PFHMC_SDDATALOW.PMSDBPCOUNT. If this was the last SD of the FPM, the value might be lower than 512. The PMSDBPCOUNT field is used by the X710-TM4 to calculate the end of the FPM space without having to read the valid bit for each individual PD entry.
 - d. Write 0 to PFHMC_SDDATALOW.PMSDTYPE
 - e. Write 1 to PFHMC_SDDATALOW.PMSDVALID
 - f. Write PFHMC_SDCMD with PMSDIDX set to the proper SD index value and PMSDWR=1

When this process completes for typical configurations, the second SD will be populated with the address of a single PD page, and entries 1-129 will be populated with address of the 128 FPM object backing pages that have been allocated.

7.9.4 De-Populating HMC Backing Pages

The process of de-populating and freeing HMC object backing pages is as follows:

- Ensure that software and hardware are not going to access objects in the pages
- Calculate the SD range and/or PD range that provide the address mapping to the X710-TM4
- Update the associated PD entries
- Invalidate the on-die PD cache entries using the PFHMC_PDINV register
- Update the SDs using the PFHMC_SDCMD, PFHMC_SDDATALOW, and PFHMC_SDDATAHIGH registers to notify the X710-TM4 that the pages are no longer valid for use.



7.9.4.1 Removing a Backing Page

As soon as software has determined that a backing page is no longer needed, the software must clear the PD_Valid bit (see [Table 7-184](#)) in the PD entry that references the backing page. After clearing the PD_Valid bit in the PD in host memory, software must then write the PFHMC_PDINV register with the SD index and PD index of the newly invalidated PD entry. This is to ensure that references to the invalid PD entry have been removed from any the X710-TM4 cache. When this write is complete, the backing page can be freed by software. The write to the PFHMC_PDINV register is not required for direct SDs since no PD is involved in addressing the HMC backing pages.

7.9.4.2 Removing a Page Descriptor Page

As soon as software has determined that an entire PD page is no longer needed, the PFHMC_SDDATALOW register must be written with PFHMC_SDDATALOW.PMSDVALID set to 0 and then the PFHMC_SDCMD must be written with PMSDIDX set to the proper SD index value and PMSDWR=1. After this sequence is complete, software is free to deallocate or re-use the PD page.

7.9.5 HMC Error Reporting

HMC related errors are reported through the PFHMC_ERRORINFO and PFHMC_ERRORDATA registers. The HMC_ERR interrupt status bit in the PFINT_ICR0 register might also deliver an interrupt for HMC errors if the interrupt is enabled in the PFINT_ICR0_ENA register. When the HMC detects an error, it sets the PFHMC_ERRORINFO.ERROR_DETECTED bit along with the relevant information in the other fields of the PFHMC_ERRORINFO and PFHMC_ERRORDATA registers. No further notification of subsequent HMC errors associated with any given PF will be issued until the current error is acknowledged by writing a 0 to the PFHMC_ERRORINFO.ERROR_DETECTED bit. [Table 7-187](#) lists the errors detected for each HMC object and the behavior associated with each error.

Table 7-187. HMC Errors

HMC Object	Error Type(s)	Error Behavior
LAN Transmit Queue	PMF Invalid, Invalid PMF Index	Not applicable to this object type.
	Invalid LAN Queue Index or HMC Object Index Too Large	Index of LAN Queue is reported in the PFHMC_ERRORDATA register. The packets associated with the queue are not transmitted.
	HMC Private Memory Address Too Large, Segment Descriptor Invalid, Segment Descriptor Too Small, Page Descriptor Invalid	The Private Memory Address of the LAN queue object is reported in the PFHMC_ERRORDATA register. Packets associated with the queue are not transmitted.
LAN Receive Queue	PMF Invalid, Invalid PMF Index	Not applicable to this object type.
	Invalid LAN Queue Index or HMC Object Index Too Large	Index of the LAN queue is reported in the PFHMC_ERRORDATA register. Packets associated with the queue are dropped.
	HMC Private Memory Address Too Large, Segment Descriptor Invalid, Segment Descriptor Too Small, Page Descriptor Invalid	The Private Memory Address of the LAN queue object is reported in the PFHMC_ERRORDATA register. Packets associated with the queue are dropped.



7.10 Admin Queues

7.10.1 Preface

The Admin queue is designed with the following goals:

- Abstract firmware interface so that firmware can be changed, whether for added functionality or bug fixes, without changing the driver. Further extension of the firmware can allow changing parts of the hardware without modifying the driver.
- Remove MMIO access from all non-essential driver paths.
- Incorporate the VF to PF and function to function mailboxes into a single, extensible interface. Shared resources should be accessed through the Admin queue.
- It is possible to write one driver that would work both on a primary function and on a virtual function.
- A low resource driver such as a pre-boot driver or an out of the box driver can use the Admin queue for limited transmit and receive.

The X710-TM4 provides the following sets of Admin queues:

- One firmware Admin queue per PF, used for software to firmware communication, exposed in the PF memory Base Address Register (BAR).
- One firmware Admin queue per VF, exposed in the VF memory BAR

Assumptions:

- Currently, firmware does not maintain context for any driver operation, except for the state of the queue itself.
- Firmware deals with one command at a time and does not start working on a new command before finishing its current task. This may change in a future version of the firmware, so the queue mechanism must allow for it today in order to avoid the need to modify the driver if this happens. firmware does however pipeline descriptor and data fetches to optimize execution latency.

7.10.2 Queue Structure

The Admin queue is comprised of a pair of Admin transmit queue and Admin receive queue. Driver commands are posted on the Admin Transmit Queue (ATQ). Firmware completes driver commands by writing back onto the command descriptor. Events that are not an immediate result of a command are written to the Admin Receive Queue (ARQ). The driver posts empty buffers to the ARQ, and the firmware fills them with events.

Both ATQ and ARQ support direct and indirect commands. The Admin queue direct command is one that fits entirely in the queue descriptor, while an extended or indirect command is one that uses an additional buffer, specified in the descriptor. When a command needs an additional external buffer, it marks the BUF flag. If the buffer contains data that the firmware needs to read, the RD flag is used. A buffer bigger than 512 bytes (AQ_LARGE_BUF) must have the LB flag set. The maximum buffer size supported in this version of the queues is 4096 bytes.

Both queues use the same descriptor structure. All descriptors and commands are defined using Little endian notation with 32 bit words. Drivers using other conventions should take care to do the proper conversions.



Table 7-188. Admin Queue Descriptor Structure (in LE 32 Order)

+3								+2								+1								+0							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Opcode								F E I E S I B U F V F C R D L B								V F E E R R C M P D D															
Return Value / VFID								Data Len																							
Cookie High																															
Cookie Low																															
Param0																															
Param1																															
Data address high																															
Data address Low																															

Table 7-189. Admin Descriptor Field Descriptions

Name	Bytes.Bits	Description
Flags.DD	0.0	Set by firmware to mark entry done
Flags.CMP	0.1	Set by firmware to mark entry as completion
Flags.ERR	0.2	Set by firmware to mark entry as an error indication
Flags.VFE	0.3	Set by firmware to mark entry as an event forwarded from a VF driver
Flags.Reserved	0.4-1.0	Reserved, must be zeroed by sender and ignored by receiver.
Flags.LB	1.1	Set by driver to indicate that indirect buffer is longer than AQ_LARGE_BUF
Flags.RD	1.2	Set by driver to indicate that firmware needs to read indirect buffer.
Flags.VFC	1.3	Set by driver to indicate command on behalf of a VF.
Flags.BUF	1.4	This command uses additional data
Flags.SI	1.5	Do not interrupt when this command completes
Flags.EI	1.6	Interrupt on error - supersedes Flags.SI in case of an error
Flags.FE	1.7	If previous command completed in error, flush this one
Opcode	2-3	command opcode (see Table 7-199)
Datalen	4-5	indirect data length in bytes (can be used for other uses if Flags.BUF is unset)
Return value/ VFID	6-7	Return value / VF ID for command or event. It is used by firmware as a return value and reflects the error code as described in Section 7.10.7 and Table 7-198 .
Cookie High	8-11	Opaque data, echoed by receiver, high half
Cookie Low	12-15	Opaque data, echoed by receiver, Low half
Param0	16-19	First general use parameter
Param1	20-23	Second general use parameter
Data Address high	24-27	indirect data pointer (can be used for other uses if Flags.BUF is unset)
Data Address low	28-31	indirect data pointer (can be used for other uses if Flags.BUF is unset)

See [Section 7.10.5](#) for details on the different command types.



7.10.2.1 Admin Queue CSRs

The Admin queues have 32 byte descriptors. They are serviced by the following registers (Table 7-190).

{PFX} denotes the register scope prefix:

- For VFs, it is VF_
- For PFs, it is PF_
- For VSIs, (PASID_MODE), it is VSI_

Except for the name prefix the PF, VF registers are exactly the same.

Table 7-190. Admin Queue Registers

Name	Width	Comment
{PFX}ATQBAH	32 bits	High bytes of ATQ base address
{PFX}ATQBAL	32 bits	Low bytes of ATQ base address, address must be 64 byte aligned
{PFX}ATQLEN	10+4 bits	ATQ length in descriptors, MSB is set for queue enable, three error bits (critical error, overflow error and VF error) are set by firmware to indicate error conditions.
{PFX}ATQH	10 bits	ATQ head pointer (firmware updates)
{PFX}ATQT	10 bits	ATQ tail pointer (Driver updates)
{PFX}ARQBAH	32 bits	High bytes of ARQ base address
{PFX}ARQBAL	32 bits	Low bytes of ARQ base address, address must be 64 byte aligned
{PFX}ARQLEN	10+4 bits	ARQ length in descriptors, MSB is set for queue enable, three error bits (critical error, overflow error and VF error) are set by firmware to indicate error conditions (see further Section 7.10.8.1)
{PFX}ARQH	10 bits	ARQ head pointer (firmware updates)
{PFX}ARQT	10 bits	ARQ tail pointer (Driver updates)

7.10.2.2 Admin Queue Interrupts

Firmware triggers an interrupt when it completes descriptors on the transmit queue or when it posts events to the receive queue. The interrupt triggered is the other interrupt and the ADMINQ cause bit is set to signals that this was the reason. (See Section 7.5.2.3).

7.10.3 Initialization

When initializing the queue, the driver must do the following:

- The driver will allocate and setup appropriately sized host memory for the queues.
- The driver must post initialized buffers to the receive queue before it can use the transmit queue. (See Section 7.10.3.1 for Receive queue element initialization).
- The driver clears the head and tail registers for each queue. head registers are {PFX}ATQH and {PFX}ARQH. Tail registers are {PFX}ATQT and {PFX}ARQT.



- Then, the driver programs the base and length registers for each queue ({PFX}ATQBAL, {PFX}ATQBAH, {PFX}ATQLEN, {PFX}ARQBAL, {PFX}ARQBAH and {PFX}ARQLEN) and sets length.enable to 1 to inform firmware that the queue is now enabled.
- For a firmware Admin queue, driver must then issue a get version admin command and check the queue and firmware major version numbers before it can use the queue for anything else. If the major versions do not match what the driver expects, the driver reports the mismatch and fails to load. For details and current queue version number, see [Section 7.10.10.1](#), get version admin command.
- For a firmware Admin queue, after the driver has verified the queue version, it sends a Driver Version command to the device. Device then sends an indication to the BMC that the PF driver is present. This is done using Host NC Driver Status Indication in NC-SI Get Link command or via the Host Network Controller Driver Status Change AEN.
- A PF driver must have at least one buffer posted to the receive queue for each VF that is currently running. (This can never be greater than the number of logical cores in the system.) This must be done before enabling VFs.

7.10.3.1 Receive Queue Element Initialization by Driver

The driver must clear any unused fields (including unused Flags) and set data pointers and data length to a mapped DMA pointer.

The driver may set the SI and the EI flags in the receive queue element. The driver must not set the FE flag on receive queue elements.

Table 7-191. Receive Queue Element - Initial Values

Name	Bytes.Bits	Value	Remarks
Flags.DD	0.0	0	Driver Must Clear
Flags.CMP	0.1	0	Driver Must Clear
Flags.ERR	0.2	0	Driver Must Clear
Flags.VFE	0.3	0	Driver Must Clear
Flags.Reserved	0.4 1.0	0	Reserved, must be zeroed by sender and ignored by receiver.
Flags.LB	1.1	0 or 1	Set by driver if buffer is longer than AQ_LARGE_BUF (512 bytes)
Flags.RD	1.2	0	Not applicable to receive queue
Flags.VFC	1.3	0	Driver Must Clear
Flags.BUF	1.4	1	Receive queue elements always have an additional buffer
Flags.SI	1.5	Driver May set	Do not interrupt when this command completes
Flags.EI	1.6	Driver May set	Interrupt on error - supersedes Flags.SI in case of error
Flags.FE	1.7	0	Driver Must Clear
Opcode	2-3		Driver Must Clear
Datalen	4-5	Buffer Len	Additional data length in bytes
Return value / VFID	6-7		Driver Must Clear
Cookie High	8-11		Driver Must Clear
Cookie Low	12-15		Driver Must Clear
Param0	16-19		Driver Must Clear



Table 7-191. Receive Queue Element - Initial Values

Name	Bytes.Bits	Value	Remarks
Param1	20-23		Driver Must Clear
Data Address high	24-27	Buffer ADDR	Indirect data pointer
Data Address low	28-31	Buffer ADDR	Indirect data pointer

The values written by the firmware when it uses the EAQ element are discussed in the following paragraphs.

7.10.4 Driver Unload and Queue Shutdown

When shutting down the Admin queue the driver (both for PF and VF) does as follows:

- Post a Queue Shutdown Admin command (0x0003) (Section 7.10.10.3). In this command, the driver will set the Driver Unloading flag if it intends to unload.
- Software must not send any additional commands on the queue until the flow is completed

and firmware does the following:

- Wait for any pending DMA transactions from firmware to be acknowledged by HW
- Close the RX queue by clearing its enable bit.
- Send a completion to the driver as usual, honoring all the interrupt control bits in the descriptor.

Software then closes the TX queue by clearing its enable bit (This is the ATQENABLE bit in the {PFX}{TYP}{DRV}_ATQLEN register).

If the driver is unloading, it issues a function reset (PFR or VFR) to the device

If the driver is unloading, firmware will inform the BMC. This is done using Host NC Driver Status Indication in NC-SI Get Link command or via the Host Network Controller Driver Status Change AEN.

- Only for a PF driver

Note: Before the Admin queues are re-enabled, software should clear the head and tail registers of the transmit and receive Admin queue.

7.10.5 Commands Description

7.10.5.1 Direct Command

7.10.5.1.1 Direct Admin Command

The template for a command that is fully contained in the descriptor and does not need an additional data buffer.



Table 7-192. Direct Admin Command Structure

Name	Bytes.Bits	Value	Remarks
Flags.DD	0.0	0	Driver Must Clear
Flags.CMP	0.1	0	Driver Must Clear
Flags.ERR	0.2	0	Driver Must Clear
Flags.VFE	0.3	0	Driver Must Clear
Flags.Reserved	0.4-1.0	0	Reserved, must be zeroed by sender and ignored by receiver.
Flags.LB	1.1	0	A direct command has no additional buffer
Flags.RD	1.2	0	A direct command has no additional buffer
Flags.VFC	1.3	0	Driver Must Clear
Flags.BUF	1.4	0	A direct command does not have an additional write buffer
Flags.SI	1.5	Driver May set	Do not interrupt when this command completes
Flags.EI	1.6	Driver May set	Interrupt on error - supersedes Flags.SI in case of error
Flags.FE	1.7	Driver May set	If set, command will be flushed if the preceding command resulted in an error
Opcode	2-3	Opcode	Command opcode (see Table 7-199)
Datalen	4-5	0	
Return value / VFID / PFID / DRID	6-7		
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the firmware into the completion of this command.
Param0	16-19		First command parameter
Param1	20-23		Second command parameter
Data Address high	24-27		Can be used for an additional command parameter
Data Address low	28-31		Can be used for an additional command parameter



7.10.5.1.2 Direct Command Completion

Table 7-193. Direct Command Completion Event Template

Name	Bytes.Bits	Value	Remarks
Flags.DD	0.0	1	Firmware must set
Flags.CMP	0.1	1	Firmware must set
Flags.ERR	0.2	0 or 1	Firmware must set only if it reporting an error
Flags.VFE	0.3	0	Firmware must clear
Flags.Reserved	0.4 -1.0	0	Reserved, must be zeroed by sender and ignored by receiver.
Flags.LB	1.1	echo	Firmware will copy value from command
Flags.RD	1.2	echo	Firmware will copy value from command
Flags.VFC	1.3	echo	Firmware will copy value from command
Flags.BUF	1.4	echo	Firmware will copy value from command
Flags.SI	1.5	echo	Firmware will copy value from command
Flags.EI	1.6	echo	Firmware will copy value from command
Flags.FE	1.7	echo	Firmware will copy value from command
Opcode	2-3	Opcode	Command opcode (see Table 7-199)
Datalen	4-5		Can be used for an additional command parameter
Return value / VFID	6-7		Firmware return value 0=no error (for error codes see Table 7-198).
Cookie High	8-11	echo	Opaque value, will be copied by the Firmware from the command
Cookie Low	12-15	echo	Opaque value, will be copied by the Firmware from the command
Param0	16-19		First command parameter
Param1	20-23		Second command parameter
Data Address high	24-27		Can be used for an additional command parameter
Data Address low	28-31		Can be used for an additional command parameter

7.10.5.2 Indirect Command

7.10.5.2.1 Indirect Admin Command

An indirect write command uses an additional DMA buffer specified in the descriptor.

The BUF flag must be set by the driver. If the buffer is larger than 512 bytes the LB flag must be set.

This version of the queue is limited to buffers up to 4096 bytes. If the command uses the buffer to pass data, the RD flag needs to be set.

**Table 7-194. Indirect Command Template**

Name	Bytes.Bits	Value	Remarks
Flags.DD	0.0	0	Driver must clear
Flags.CMP	0.1	0	Driver must clear
Flags.ERR	0.2	0	Driver must clear
Flags.VFE	0.3	0	Driver must clear
Flags.Reserved	0.4-1.0	0	Reserved. Must be zeroed by sender and ignored by receiver.
Flags.LB	1.1	0 or 1	Set by driver if buffer is longer than AQ_LARGE_BUF (512)
Flags.RD	1.2	0 or 1	Set by driver to indicate that firmware needs to read indirect buffer.
Flags.VFC	1.3	0	Driver must clear.
Flags.BUF	1.4	1	This command uses additional data buffer. Driver must set this flag on an indirect command.
Flags.SI	1.5	Driver might set	Do not interrupt when this command completes.
Flags.EI	1.6	Driver might set	Interrupt on error - supersedes Flags.SI in case of error.
Flags.FE	1.7	Driver might set	If set, command will be flushed if the preceding command resulted in an error.
Opcode	2-3	Opcode	Command opcode (see Table 7-199).
Datalen	4-5	Buffer LEN	Usable length of additional buffer in bytes.
Return value / VFID	6-7		
Cookie High	8-11	Cookie	Opaque value. Copied by the firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by the firmware into the completion of this command.
Param0	16-19		First command parameter
Param1	20-23		Second command parameter
Data Address High	24-27	Buff Addr	High bits of buffer address
Data Address Low	28-31	Buff Addr	Low bits of buffer address

7.10.5.2.2 Indirect Command Completion

When completing an indirect command the firmware will overwrite the datalen with the actual length of data returned by the command.

Table 7-195. Indirect Command Completion Template

Name	Bytes.Bits	Value	Remarks
Flags.DD	0.0	1	Firmware must set
Flags.CMP	0.1	1	Firmware must set
Flags.ERR	0.2	0 or 1	Firmware must set only if it reporting an error
Flags.VFE	0.3	0	Firmware must clear
Flags.Reserved	0.4 -1.0	0	Reserved. Must be zeroed by sender and ignored by receiver.



Table 7-195. Indirect Command Completion Template (Continued)

Name	Bytes.Bits	Value	Remarks
Flags.LB	1.1	echo	Firmware copies value from command
Flags.RD	1.2	echo	Firmware copies value from command
Flags.VFC	1.3	echo	Firmware copies value from command
Flags.BUF	1.4	echo	Firmware copies value from command
Flags.SI	1.5	echo	Firmware copies value from command
Flags.EI	1.6	echo	Firmware copies value from command
Flags.FE	1.7	echo	Firmware copies value from command
Opcode	2-3	Opcode	Command opcode (see Table 7-199)
Datalen	4-5		Can be used for an additional command parameter
Return value / VFID / PFID / DRID	6-7		Firmware return value 0=no error (for error codes see Table 7-198)
Cookie High	8-11	echo	Opaque value. Copied by the firmware from the command.
Cookie Low	12-15	echo	Opaque value. Copied by the firmware from the command.
Param0	16-19		First command parameter
Param1	20-23		Second command parameter
Data Address High	24-27		Can be used for an additional command parameter
Data Address Low	28-31		Can be used for an additional command parameter

7.10.6 Firmware Command Fetch and Verification

When a command is posted, firmware looks it up in an internal permission table to decide if the request should be honored. Possible actions are:

- Allow - Firmware acts upon the command.
- Forward - Firmware halts the queue and forward the command to the PF driver, a completion for this command is initiated by the PF driver when it finishes it. Only then is further processing of commands from this queue allowed. (The PF driver needs to re-enable the queue after it deals with the command.)
- Error - Firmware completes the action by returning the error specified in the table.
- Drop - Firmware behaves as if the command succeeds, but does nothing.

A VF driver is only allowed to post the commands listed in Table 7-196. If any other command is posted by a VF, firmware returns the EPERM (operation not permitted) error code.

Table 7-196. Commands Allowed for VF Firmware Admin Queues

Opcode	Command	Ref
0x0001	Get Version	Section 7.10.10.1
0x0801	Send to PF	Section 7.10.11.1



7.10.7 Error Handling

When firmware encounters an error it uses the return value field to indicate the type of error. The error code is comprised of two bytes: the lower byte is a user-visible code from [Table 7-198](#). The higher byte may be used by firmware to report internal state or debug information. The driver must log this information. Firmware may change this value from release to release. It is not to be reported to the user and no other action is to be taken upon this data. When the return value is 0 (no error), firmware must not set the high byte.

Table 7-197. Admin Queue Return Value Fields

Name	Bytes	Comment
Code	0:7	Return value from Table 7-198
Firmware internal Code	8:15	Firmware internal code, must be 0 if Code field is 0

7.10.8 Error Codes

When Firmware completes a command it must use the error codes in [Table 7-198](#).

Table 7-198. Admin Queue Return Values and Error Codes

Error Code	Value	Meaning
(No Error)	0	No Error (success)
EPERM	1	Operation not permitted
ENOENT	2	No such element
ESRCH	3	Bad opcode
EINTR	4	Operation interrupted
EIO	5	I/O error
ENXIO	6	No such resource
E2BIG	7	Arg too long
EAGAIN	8	Try again
ENOMEM	9	Out of memory
EACCES	10	Permission denied
EFAULT	11	Bad address
EBUSY	12	Device or resource busy
EEXIST	13	Attempt to create something that exists
EINVAL	14	Invalid argument
ENOTTY	15	Not a typewriter
ENOSPC	16	No space left or allocation failure
ENOSYS	17	Function not implemented
ERANGE	18	Parameter out of range
EFLUSHED	19	Command flushed because a previous command completed in error

**Table 7-198. Admin Queue Return Values and Error Codes**

Error Code	Value	Meaning
BAD_ADDR	20	Internal error, descriptor contains a bad pointer
EMODE	21	Operation not allowed in current device mode
EFBIG	22	File too big

7.10.8.1 Critical Error Indication

- On any error that prevents data placement to a queue, the ATQLEN.ATQCRIT (or ARQLEN.ARQCRIT) bit is set by firmware and the queue is stopped (by clearing its enable bit), then an interrupt is sent by firmware to the driver.
- Software reads and reports the error code, and then resets the queue.
- If an overflow occurs and a message to the queue is dropped because the queue is full, firmware sets ARQLEN.ARQOVFL and interrupts the driver. (Firmware does not stop the queue since, depending on the driver mode, this may be a recoverable error.)
- Note: This error can currently only happen on the receive queue, but to simplify the hardware design the bit is present on both queues.
- When a VF has an event that causes firmware to set an error bit in its receive queue, firmware sets ARQLEN.ARQVFE in the corresponding PFs queue and interrupts it. (Firmware does not stop the PF queue in this case.)

Note: Events and completions that have already been posted before the error are still readable and can be handled by software.

7.10.9 Command Opcodes

Opcodes are 16 bits. The upper 8 designate the group of the opcode. The lower 8 are the command in the group. Each group is described in its relevant chapter.

Table 7-199. Opcode Groups

Name	Opcodes	Ref/Remark
Generic	0x00XX and 0x0206	Section 7.10.10 below and Table 7-200
Mac Address	0x0100	Section 4.2.1.5
PXE Mode	0x0110	Section 8.2.2.1
Switch	0x02XX	Section 7.4.9.5 and Table 7-80
DCB	0x03XX	Section 7.7.5
Scheduler	0x04XX	Section 7.8.4 and Table 7-185
Link	0x06XX	Section 3.2.10
NVM	0x07XX	Section 3.4.10
Virtualization	0x08XX	Section 7.10.11



Table 7-199. Opcode Groups (Continued)

Name	Opcodes	Ref/Remark
Alternate structure	0x09XX	Section 4.3.2.2
LLDP	0x0AXX	Section 7.12.5.2.3
Firmware internal and firmware tests	0xFFXX	

7.10.10 Generic Firmware Admin Commands

Table 7-200. Generic Commands

Name	Opcode	Ref	Type
Get Ver	0x0001	Section 7.10.10.1	direct
Driver Version	0x0002	Section 7.10.10.2	indirect
Queue Shutdown	0x0003	Section 7.10.10.3	indirect
Set PF context	0x0004	Section 7.10.10.4	direct
Get Expanded AQ Error Reason	0x0005	Section 7.10.10.5	direct
Request resource ownership	0x0008	Section 7.10.10.5	direct
Release resource ownership	0x0009	Section 7.10.10.6	direct
Discover function capabilities	0x000A	Section 7.10.10.7	indirect
Discover device capabilities	0x000B	Section 7.10.10.7	indirect
CSR Access	0x0206	Section 7.10.10.8	direct

7.10.10.1 Get Version Admin Command

This must be the first command that the driver issues before it can use the queue for other purposes. The driver must inspect the reply to ensure that the firmware version is compatible. If firmware is still initializing it may delay response until it completes. Both the firmware and the API have two unassigned 16 bit values as minor and major version. The driver must not continue loading if the major version does not match. Minor versions are for tracking changes that do not need driver modifications.

Table 7-201. Get Version Command (Opcode: 0x0001)

Name	Bytes.Bits	Value	Remarks
Flags	1-0		See Section 7.10.5.1.2 for details.
Opcode	2-3	0x0001	Command opcode
Datalen	4-5	0	No external response buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command



Table 7-201. Get Version Command (Opcode: 0x0001)

Name	Bytes.Bits	Value	Remarks
ROM build ID	16-19		Device ROM build version.
Firmware build ID	20-23		Device firmware build version.
Firmware Major Version	24-25		Major firmware version (unsigned 16 bit integer).
Firmware Minor Version	26-27		Minor firmware version (unsigned 16 bit integer).
API major ver	28-29	1	Major queue version (unsigned 16 bit integer) 1 for this version of the queues.
API minor ver	31-32	1	Minor queue version (unsigned 16 bit integer) 1 for this version of the queues.

7.10.10.2 Driver Version Indirect Admin Command

This command is used by the driver to report the driver version. The firmware should use the driver version in order to report it to the BMC. See [Section 7.10.3](#).

Table 7-202. Driver Version Command (Opcode: 0x0002)

Name	Bytes.Bits	Value	Remarks
Flags	1-0		See Section 7.10.5.1.2 for details.
Opcode	2-3	0x0002	Command opcode
Datalen	4-5		Buffer Length. Can be up to 32.
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Driver version	16-19	version	byte 16 = major version byte 17 = minor version byte 18 = build version byte 19 = sub-build version
Reserved	20-23	0x0	Reserved
Data Address high	24-27		Address of response buffer
Data Address low	28-31		Address of response buffer

Table 7-203. Driver Version Buffer

Name	Length	Description
Driver version	Datalen	Driver string (not null terminated) as reported by driver:



7.10.10.3 Queue Shutdown command

This is the final command posted to the queue, closing the queue as described in [Section 7.10.4](#). After this command completes, the driver is allowed to free any host resources associated with the Admin queue.

If the driver is going to unload, it must set the Driver Unloading flag to inform firmware.

As soon as this command is posted, the driver is not allowed to issue any more commands on the queue before a reset is done.

Note: Interrupt generation and the interrupt control flags in this command are handled as usual by firmware. This means that if an interrupt was not inhibited by setting the SI flag, it will happen. If the driver is in polling mode and cannot handle an interrupt, it needs to either inhibit the interrupt or have interrupts disabled through the interrupt control registers.

Table 7-204. Queue Shutdown Command (Opcode: 0x0003)

Name	Bytes.Bits	Value	Remarks
Flags	1-0		See Section 7.10.5.1.2 for details.
Opcode	2-3	0x0003	Command opcode
Datalen	4-5	0	No external data
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Driver unloading	16.0		1 if the driver intends to unload, 0 otherwise
Reserved	16.1-16.7 17-31	0x0	Reserved

7.10.10.4 Set PF Context

This admin command is used to set explicit PF ID number. It is needed for some admin commands that requires control on specific PF context, regardless of the PF from which the admin command is initiated.

Table 7-205. Set PF Context Command (Opcode: 0x0004)

Name	Bytes.Bits	Value	Remarks
Flags	1-0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0004	Command opcode
Datalen	4-5	0	No external data
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
PF ID	16		Physical function ID
Reserved	17-31	0x0	Reserved



7.10.10.5 Request Resource Ownership Admin Command

This command is used by the driver to request ownership of a shared resource. The driver specifies the resource and the type of access it requests (listed in [Table 7-206](#)). On success the command returns a return value of 0. The completion specifies the maximum time in ms that the driver may hold the resource in the Timeout field. The driver must free the resource before that time. The driver must free a resource before asking for it again. A request for a resource that is already held by this driver will fail with the EACCESS error code. A timeout value of zero means no timeout.

If the resource is held by someone else and the resource is different from Global Config Lock, the command completes with a EBUSY return value and the timeout field indicates the maximum time the current owner of the resource has to free it. For the Global Config Lock resource, the command always completes successfully, while the status is reflected in the Status field of the command.

Firmware implements a timeout mechanism taking back the ownership if the driver hangs. Any further commands by this driver that attempt to access this resource will fail with the EPERM error code until the driver frees the resource and requests it again.

Table 7-206. Shared Resources

Resource	ID	Supported modes	Default Timeout
NVM	0x0001	1=read, 2=write	3000ms for read, 180000ms for write
SDP	0x0002	1=read, 2=write	0 (no timeout)

Table 7-207. Request Resource Ownership Admin Command (Opcode: 0x0008)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0008	Command opcode
Datalen	4-5	0	No external buffer for this command
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by the firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by the firmware into the completion of this command
Resource ID	16-17	ID	see Table 7-206 for list of IDs
Access Type	18-19	Type	see Table 7-206 for list
Timeout	20-23	timeout	Timeout in ms. This field is returned in the firmware response structure and indicates the timeout used for the specific resource.
Resource number	24-27	Number	For an SDP, this is the pin ID of the SDP.
Status	28-29	Status	Written by firmware. Used for Global Config Lock (Resource ID 0x0004), reserved for others 0x0000 – Global Config Lock acquired successfully (STAT_SUCCESS) 0x0001 – Global Config Lock acquired by another driver, configuration is in progress (STAT_INPROGRESS) 0x0002 – Global Lock could not be acquired, configuration was already completed (STAT_COMPLETED)
Reserved	30-31		



7.10.10.6 Release Shared Resource Admin Command

This command is used to return ownership of a shared resource to the firmware. The driver will specify the ID of the resource it is releasing in the ID field.

Table 7-208. Release Resource Ownership Admin Command (Opcode: 0x0009)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0009	Command opcode
Datalen	4-5	0	No external buffer for this command
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Resource ID	16-17	ID	see Table 7-206 for list of IDs
Reserved	18-23	Reserved	Reserved
Resource number	24-27	Number	If a resource number is specified in the request, it needs to be specified here too.
Reserved	28-31	Reserved	Reserved

7.10.10.7 Discover Device/Function Capabilities

This command is used to request the list of capabilities of the device or the function. The firmware will fill in the capabilities structure and return the length to the driver. If the buffer size is not big enough for the whole structure, the firmware will return ENOMEM (at the return value), write the number of capabilities that should be returned by the command to the cap count field, and reset the length field to zero. This can be used to discover the number of capabilities returned, and the structure size can be calculated by software using the known size of the capability structure. Capabilities are described using the structure in [Table 7-209](#). The list of resources recognized by this version of the command are in [Table 7-210](#). Additional capabilities can be retrieved using the Get PHY Abilities (0x0600) command and the Get Switch Resource Allocation (0x0204) command (see [Section 7.4.9.5.3.5](#)).

Note: Unsupported capabilities are not reported.

Table 7-209. Discover Device/Function Capabilities (Opcode: 0x000A, 0x000B)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x000A or 0x000B	Command opcode 0x000A for function capabilities 0x000B for device
Length	4-5	buffer length	Length of buffer
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Reserved	16-19	Reserved	



Table 7-209. Discover Device/Function Capabilities (Opcode: 0x000A, 0x000B)

Name	Bytes.Bits	Value	Remarks
cap count	20-23		Number of capability records returned (zeroed by driver written by firmware).
Data Address high	24-27	0x0	Address of response buffer
Data Address low	28-31	0x0	Address of response buffer

Table 7-210. Capability Structure

Name	Length	Remarks
Capability	16 bits	See Table 7-209 for list of capabilities
Major Version	8 bits	
Minor Version	8 bits	
Number	32 bits	Number of resources described by this capability
Logical Id	32 bits	Only meaningful for some types of resources
Physical ID	32 bits	Only meaningful for some types of resources
Reserved1	64 bits	Reserved
Reserved2	64 bits	Reserved

Table 7-211. Resources Recognized by This Version of the Command

Name	Capability	Ver	Number	Logical ID	Physical ID
Switching mode	0x0001	1.0	Returns the Switching mode supported: 0: EVB switching (including cloud) 6: L4 Port Filters Mode 1 7: L4 Port Filters Mode 2 8: L4 Port Filters Mode 3 Other: Reserved	Channel Identifier: 0x0 S-tag 0x1 VLAN	
Manageability Mode	0x0002	2.0	Bits 3:0 Returns the value of the Manageability Pass Through Mode field in the NVM Bits 7:4: Returns the value of the Control Interface field in the NVM Bits 11:8: Returns the value of the Redirection Sideband Interface field in the NVM Bits 31:12: Reserved	Supported Protocols over MCTP Bit 0: Reserved Bit 1: PLDM Bit 2: OEM commands Bit 3: NC-SI Bit 31:4: Reserved	
OS2BMC Capable	0x0004	1.0	Returns the value of the OS2BMC Capable field in the NVM		
Functions Valid	0x0005	1.0	Bits 15:0 each corresponds to a PFPCI_STATUS1.FUNC_VALID bit for PFs 0,...,15		
Alternate RAM Structure	0x0006	1.0	1		



Table 7-211. Resources Recognized by This Version of the Command (Continued)

Name	Capability	Ver	Number	Logical ID	Physical ID
WoL and Proxy support	0x0008	1.0	Number of ACPI supported filters = 8 if supported, 0 otherwise. Same value for all ports and for the device command. For the device return the total number of filters across all supporting functions.	SEID of WoL and Proxy VSI of the port. N/A if programming method is HW. N/A for device.	Bit0: APM WoL is supported (based on PFPM_APM.APME bit For Device command - OR of all the PFs bits. Bit 1: ACPI Programming Method (relevant only if Number of ACPI supported filters is not zero) 0 = HW 1 = AQC Bit 2: Proxy Support 0 = Disabled 1 = Enabled For Device command - OR of all the PFs bits. Other bits - reserved
SR-IOV	0x0012	1.0	Set to one if enabled in config space For device should be set if set in any of the functions.	SR-IOV version (1.1)	
Virtual Function	0x0013	1.0	Function: Number of allocated VFs Device: Total number of VFs exposed to all functions	Logical ID of first VF	
VMDq	0x0014	1.0	Set to one		
802.1Qbg	0x0015	1.0	One if enabled		
VSI	0x0017	1.0	Function: Number of guaranteed VSI As read from PF allocations structure for PFs Device: Number of VSIs allocated to the host (not including EMP VSIs).		
DCB	0x0018	1.0	One if enabled	Device = 0: Bitmap of active TCs	Max number of TCs (8 for this product)
iSCSI	0x0022	1.0	0x1 if iSCSI is enabled. 0x0 if not enabled. For a device capability it is always set, and for a function capability it is a reflection of the PFGEN_STATE.PFSCEN flag.		
RSS	0x0040	1.0	Table size 512for PFs. 64 for VFs.	Entry width in bits 6 for PFs. 4 for VFs.	
RX Queues	0x0041	1.0	Function: Number of queues allocated to the PF. Device: Total number of queues available to the device		ID of first queue



Table 7-211. Resources Recognized by This Version of the Command (Continued)

Name	Capability	Ver	Number	Logical ID	Physical ID
TX Queues	0x0042	1.0	Function: Number of queues Device: Total number of queues available.		ID of first queue
MSI-X	0x0043	1.0	Number of vectors		
VF-MSIX	0x0044	1.0	Number of MSIX Vectors available to VFs of this PF		
Flow Director	0x0045	1.0	Function: Number of filters guaranteed to this PF. Device: Number of filters available in device (8192)	Function: Number of best effort filters Device: Number of filters available in device (8192)	
1588	0x0046	1.1	Function: TimeSync is enabled on the port on which this function runs Device: TimeSync enabled on any of the ports	Function: N/A Device: A bitmap of the enabled ports	
MaxMTU	0x47	1.0	For function: Max MTU of the function. For device: Max MTU of the HW (9728)		
NVM versions ¹	0x48	1.0	1 st word – NVM word address, 2 nd word – NVM value	1 st word – NVM word address, 2 nd word – NVM value	1 st word – NVM word address, 2 nd word – NVM value
LED ²	0x0061	1.0	Always 1	Action	Pin number (GPIO Index)
SDP ³	0x0062	1.0	Always 1	Action	Pin number (GPIO Index)
MDIO ⁴	0x0063	1.0	One if enabled	Mode: 0x0: MDIO Shared 0x1: MDIO dedicated 0x2: i2c	The interface used to control this port.
WR_CSR_PROT	0x0064	1.0	Bytes 0 ... 3 of the WR_CSR_PROT parameter. LS byte is byte 0 of the field.	Bytes 4 ... 7 of the WR_CSR_PROT parameter. MS byte is byte 7 of the field.	

1. The NVM Versions capability is used to return NVM version data to software in a centralized location. It will appear multiple times, reporting three NVM versions per entry. The following NVM words are returned as versions: 0x18-0x2E (inclusive) and 0x34, 0x35. An NVM word address of 0xFFFF indicates an invalid field.
2. Repeat this entry for each assigned LED. . If Device = 1, returns an entry for each LED in the device.
3. Repeat this entry for each assigned pin.If Device = 1, returns an entry for each SDP in the device.
4. This entry is not relevant for device = 1 and is not returned.

Note: Fields that are not applicable (empty in the table) are set to zero by the firmware.

Note: When device capabilities are requested total numbers for the whole device should be returned.



7.10.10.8 Receive Control CSR Read / Write Admin Command

This command should be used when accessing the following registers. Accessing this command using the PF driver, software can access all registers listed in [Table 7-212](#). Accessing this command using the VF driver, software can only access the VFQF_HENA, VFQF_HREGION and VFQF_HKEY registers.

[Table 7-212](#) lists the registers that should be accessed by the Receive Control CSR Read / Write Admin Command rather than by direct access.

Table 7-212. Receive Control Registers List Accessed by Admin Command

CSR Name	CSR Name	CSR Name	CSR Name	CSR Name	CSR Name
PFQF_CTL_0	PFQF_FDALLOC	PFLAN_QALLOC	PFQF_HREGION	PFQF_HENA	VPQF_CTL
VFQF_HENA	VFQF_HREGION	VSILAN_QTABLE	VSILAN_QBASE	VSIQF_CTL	VSIQF_TCREGION
PFQF_HKEY	VFQF_HKEY	PRTQF_CTL_0	GLQF_CTL	GLQF_SWAP	GLQF_HASH_MSK
GLQF_FD_MSK	PRTQF_FD_INSET	PRTQF_FD_FLXINSET	PRTQF_FD_MSK	GLQF_HSYM	GLQF_HASH_INSET

Table 7-213. Receive Control CSR Read Command (Opcode: 0x0206)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0206	Command opcode
Length	4-5	0x0	
Return value/VFID	6-7		Set to 0x0 by the software at command initiation. Set by the device to the following values at command completion: 0 No Error EACCES - Access denied (attempt to access to non-permitted CSR) EAGAIN - Access failed (Software may try the command once again)
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Reserved	16-19		
Address	20-23		Register Address
Reserved	24-27	0x0	Reserved
Data	28-31	0x0	Set to 0x0 by software. Returned register value by the device.

Table 7-214. Receive Control CSR Write Command (Opcode: 0x0207)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0207	Command opcode
Length	4-5	0x0	



Table 7-214. Receive Control CSR Write Command (Opcode: 0x0207) (Continued)

Name	Bytes.Bits	Value	Remarks
Return value/VFID	6-7		Set to 0x0 by the software at command initiation. Set by the device to the following values at command completion: 0 No Error EACCES - Access denied (attempt to access to non-permitted CSR) EAGAIN - Access failed (Software may try the command once again)
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command
Reserved	16-19		
Address	20-23		Register Address
Reserved	24-27	0x0	Reserved
Data	28-31	0x0	Register value programmed by software.

7.10.11 Virtualization Admin Commands

Table 7-215. Virtualization Admin Commands

Name	Opcode	Ref	Type
Send to PF	0x0801	Section 7.10.11.1	Indirect /Direct
Send to VF	0x0802	Section 7.10.11.2	Indirect /Direct

7.10.11.1 Send Message to PF Admin Command

This command, together with the next one, implement a communication channel between PFs and their VFs. The data in the external buffer is copied into an event on the PF receive queue. The command completes once the data is copied. The contents of the messages will be defined by software.

Since the value of the cookie is copied to the event, if 8 bytes are enough for needed message, the driver may specify a length of zero, and not use an external buffer. In this case it should also not set the BUF flag.

Note: This version of the queues supports messages of up to 4096 bytes.

Table 7-216. Send to PF Command (Opcode: 0x0801)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0801	Command opcode
Length	4-5	buffer length	Length of message
Return value/VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.



Table 7-216. Send to PF Command (Opcode: 0x0801) (Continued)

Name	Bytes.Bits	Value	Remarks
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-19		
Reserved	20-23		
Data Address high	24-27	0x0	Address of data buffer
Data Address low	28-31	0x0	Address of data buffer

After posting the event to the PF Admin receive queue, the firmware will complete this command by updating the flags and return value (0 for success).

Table 7-217. Message from VF Event (Opcode: 0x0801)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0801	Event code
Length	4-5	buffer length	Length of message
Return value/VFID	6-7	VFID	ID of sending VF.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware from the source descriptor.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware from the source descriptor.
Reserved	16-19		
Reserved	20-23		
Data Address high	24-27	0x0	Address of data buffer
Data Address low	28-31	0x0	Address of data buffer

7.10.11.2 Send Message to VF Admin Command

This command, together with the previous one, implements a communication channel between PFs and their VFs. The data in the external buffer is copied into an event on the VF receive queue. The command completes once the data is copied. The contents of the messages are defined by software. A PF can only send messages to one of its VFs.

Since the value of the cookie is copied to the event, if 8 bytes are enough for needed message, the driver may specify a length of zero, and not use an external buffer. In this case it should also not set the buf flag.

Note: This version of the queues supports messages of up to 4096 bytes.

Table 7-218. Send to VF Command (Opcode: 0x0802)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0802	Command opcode
Length	4-5	buffer length	Length of message
Return value	6-7		Return value. Zeroed by the software device driver. Written by firmware.

**Table 7-218. Send to VF Command (Opcode: 0x0802) (Continued)**

Name	Bytes.Bits	Value	Remarks
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
VFID	16-19	VFID	ID of VF
Reserved	20-23		
Data Address high	24-27	0x0	Address of data buffer
Data Address low	28-31	0x0	Address of data buffer

After posting the event to the VF Admin receive queue, the firmware will complete this command by updating the flags and return value (0 for success).

Table 7-219. Message from PF Event (Opcode: 0x0802)

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.1.1 for details.
Opcode	2-3	0x0802	Event code
Length	4-5	buffer length	Length of message
Return value/VFID	6-7		reserved
Cookie High	8-11	Cookie	Opaque value. Copied by firmware from the source descriptor.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware from the source descriptor.
Reserved	16-19		
Reserved	20-23		
Data Address high	24-27	0x0	Address of data buffer
Data Address low	28-31	0x0	Address of data buffer

7.10.12 Debug Admin Command for Thermal Sensors Programming

This admin command is used to read or modify the thermal threshold levels of the sensors on the board. It is based on the Thermal Sensors Header structure in the NVM to identify the existence of thermal sensors (TS_CNT > 0).

Note that the programmed thermal threshold values do not persist across power cycle nor across global or EMP resets.

The Thermal Sensors Programming admin command is enabled from the PF as well as the tools admin queues.

**Table 7-220. Thermal Sensors Programming Admin Command (0x0721)**

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0721	Command opcodes.
Data Length	4-5	0x0	
Return Value/ VFID	6-7	0x00	N/A (reserved zero).
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Sensors Action	16	0x2	0x0 = Read current setting. 0x1 = Modify thermal sensors parameters. 0x2 = Read PHY temperature. Else = Reserved.
Reserved	17 - 23		Reserved.
Data Address High	24 - 27		Address of the data buffer. This field is meaningful only if the <i>Select Data</i> field in this command is set to 0x1.
Data Address Low	28 - 31		

Table 7-221. Completion of the Set Thermal Threshold Admin Command (0x0721)

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0721	Command opcode.
Data Length	4-5	0x0	
Return Value/ VFID	6-7		Command Error Code. 0x0 = No error. EPERM = The thermal sensors structure is not found in the NVM. EIO (0x5) = Programming failed due to NACK on the I ² C bus or sequence fault.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Sensors Action	16	0x2	Returns the action that was requested by the Thermal Sensors Programming command: 0x0 = Read current setting. 0x1 = Modify thermal sensors parameters. 0x2 = Read PHY temperature. Else = Reserved.
Reserved	17-23		Reserved
Data Address High	24 - 27		Address of the data buffer. This field is meaningful only if the <i>Select Data</i> field in this command is set to 0x1.
Data Address Low	28 - 31		





7.11 Statistics

The X710-TM4 provides statistics for the following interfaces:

- Physical ports
- Virtual switch elements: VEBs, PVs, and VSIs
- Flow director statistics.
- Host interface statistics

A minimal set of Ethernet interface group statistics (RFC 2863) is provided by The X710-TM4 at the switch sampling points, a fuller set of statistics is provided for physical ports.

The prefixes for the statistics counter names are listed in [Table 7-222](#).

Table 7-222. Counter Name Prefixes

Element	Register Prefix	Instances	Remark
Port	GLPRT	4	
VEPA or VEB	GLSW	16	
VEB per VLAN	GLVEBVL	128	
VEB per UP	GLVEBTC	16x8	Two dimensional array of 8 UPs for 16 VEBs
VSI	GLV	384	

7.11.1 Counter Implementation

Most of the counters in the X710-TM4 are used by more than one entity, for example VSI counters are both a virtual switch port (hence used by a switch monitor agent) and a driver interface. the X710-TM4 does not provide clear-on-read statistics counters, since these would need to be duplicated per client.

As a general rule, the X710-TM4 statistics counters should not be reset by software, because they are shared between more than one owner. Software needs to maintain a delta from the first value seen. If a software device driver needs the ability to reset counters, this should be implemented by updating the delta reference value.

To allow Firmware to reset counters when a switching element is allocated, or for debug, the counters are implemented as clear on write, that means that writing any value clears the counter.

The X710-TM4 provides 48-bit counters for byte and packet counters, and 32 bit counters for errors and discard events. Software should maintain counters that are appropriate in size for the OS and MIB requirement.

48 bit counters are implemented as two registers whose names end with high and low containing the upper 16 bits and lower 32 bits respectively.

When accessed using a 64 bit operation from PCI these accesses are guaranteed by design to be atomic. (They are internally converted to two 32 bit accesses that are always consecutive with no interleaving between reads from different drivers. A special hardware indication tells the statistics block that this is the case. The statistics block performs one read of the counter memory to satisfy both requests, thus providing atomicity.)



When 32 bit accesses are used to read the counters each of them cause a separate read from the statistics block and therefore care must be taken to properly handle the possibility of one half around between the reads, since atomicity is not guaranteed.

VFs have no access to statistics registers, they should query the PF through the VF to PF virtual channel for their statistics.

7.11.2 Statistics Consistency Rules

- At each switch sample point, a packet is either good or error. That means that when a packet is discarded it is not counted in any of the none-error counters. It is also never counted by any of the next processing stages.
- All drops are counted. A packet with an error is only counted in one counter. An exception to this rule is that some debug counters are not mutually exclusive with error counters, for example GLPRT_ERRBC and GLPRT_MSPDC are also counted as CRC errors and undersized errors respectively.
- Ethernet octet counters count the packet as seen on the wire, from the Ethernet header up to and including the CRC (for both RX and TX). This means that even though an RX packet might be stripped of tags before placement the byte counters record the original size. TX packets are counted after all offloads and insertions.
- Flow control packets are only counted in the flow control counters.
- Statistics specific to count packets as they are delivered to the offload engine on RX, and as they leave the engine on TX. This applies to both packet lengths and packets that were merged or split. For example packets merged by LRO are counted before the aggregation.
- Maximum lengths are adjusted to accommodate added tags. For example the highest histogram bin MIB counter GLPRT_PTC9522 will also count any packet that would have fit in it without the added S-TAGs and VLANs.

7.11.3 Supported MIBs

The X710-TM4 supports different statistic counters as described in this chapter. The statistic counters can be used to create statistic reports as required by different standards. The X710-TM4 statistic counters allow support for the following standards:

- IEEE Std. 802.3-2018 clause 30 management – DTE section.
- NDIS 6.0 OID_GEN_STATISTICS.
- RFC 2819 – RMON Ethernet statistics group, for ports and VSIs.
- RFC 2863 – SMON Ethernet statistics group, for various switch elements.
- Linux Kernel
- net_device_stats

The following section describes the match between the internal the X710-TM4 statistic counters and the counters requested by the different standards.



7.11.4 Interface Statistics at VSIs and logical Interfaces

The X710-TM4 provides Ethernet interface group statistics per RFC 2863 on each of the Virtual Station Interfaces (VSI), Port aggregators and Virtual bridges. [Table 7-223](#) provides an illustration of interface counters in the X710-TM4 and their sizes.

VF and PF VSIs have the same counter set. The only difference between the counter set for VSIs and the one for other switch element is that the RUPP (unknown protocol) counter is replaced by a missed packet counter RMPC.

Note that the counter names repeat themselves with a different prefix for the element type. See [Table 7-223](#) for the list of prefixes. The width of the counters that are provided, is calculated to allow ample time for software to get the statistics before they wrap around. The 48-bit counters are a pair of registers, one ending with an L that holds with the lower 32-bits of the counter, the higher 12 bits rest of the bits are in a register ending with and H (denoted in [Table 7-223](#) as {H,L}). Not all counters are implemented for all prefixes.

Table 7-223. Ethernet Interface Group Statistics Counters

Register Name	Width	Description
[PFX]GORC{H,L}	48	Incoming packet octets
[PFX]UPRC{H,L}	48	Incoming number of unicast packets
[PFX]MPRC{H,L}	48	Incoming number of multicast packets
[PFX]BPRC{H,L}	48	Incoming number of broadcast packets
[PFX]RDPC	32	Incoming packet discards
[PFX]RUPP	32	Incoming unknown protocol packets
[PFX]GOTC{H,L}	48	Outgoing packet octets
[PFX]UPTC{H,L}	48	Outgoing number of unicast packets
[PFX]MPTC{H,L}	48	Outgoing number of multicast packets
[PFX]BPTC{H,L}	48	Outgoing number of non unicast packets
[PFX]TDPC	32	Outgoing number of discards
[PFX]TEPC	32	Outgoing packet errors

Some error counters are not implemented at specific switch elements because they cannot happen at those elements in the current switch design. If software is queried about the value of these counters it should return zero (see [Table 7-224](#)).

Table 7-224. Error Counters Not Implemented in Current Design

Counter Name	Switch element	Description
GLPRT_XEC	Port	Port XSUM Error Count
GLPRT_TEPC	Port	Port transmit error packet count (GPLRT_TDOLD counts drops due to link down)
GLV_TDPC	VSI	VSI transmit packets discarded count



Table 7-224. Error Counters Not Implemented in Current Design

Counter Name	Switch element	Description
GLSW_RDPC	PA or VEB	Switch receive packets discarded count (Replaced by GLSWID_RUPP)
GLSW_TEPC	PA or VEB	Switch transmit error packet count
GLPRT_TDPC	Port	Packets that were discarded on transmit while link was down.

7.11.4.1 MAC or Physical Up-link Interface Statistics

The MAC or Physical up-link interface statistics counters are accessible to the device management/control entity in VMM or IOVM through the PF. The MAC or Physical link interface statistics use the counters defined in Table 7-225. The only exception is the Unknown Protocol Packet counter, which has no meaning in the case of the physical port and therefore is absent.

In addition, the X710-TM4 provides per MAC port some of the statistics out of the IEEE Std. 802.3-2018 clause 30 management counters as well as part of the RMON Ethernet statistics group as defined by IETF RFC 2819. See Table 7-225.

Table 7-225. Additional Per-Port Counters

Register Names	Width	Description
GLPRT_PRC64{H,L}	48	Packets Received [64 Bytes] Count
GLPRT_PRC127{H,L}	48	Packets Received [65–127 Bytes] Count
GLPRT_PRC255{H,L}	48	Packets Received [128–255 Bytes] Count
GLPRT_PRC511{H,L}	48	Packets Received [256–511 Bytes] Count
GLPRT_PRC1023{H,L}	48	Packets Received [512–1023 Bytes] Count
GLPRT_PRC1522{H,L}	48	Packets Received [1024 to 1522] Count
GLPRT_PRC9522{H,L}	48	Packets Received [1523 to Max Bytes] Count
GLPRT_PTC64{H,L}	48	Packets Transmitted (64 Bytes) Count
GLPRT_PTC127{H,L}	48	Packets Transmitted [65–127 Bytes] Count
GLPRT_PTC255{H,L}	48	Packets Transmitted [128–255 Bytes] Count
GLPRT_PTC511{H,L}	48	Packets Transmitted [256–511 Bytes] Count
GLPRT_PTC1023{H,L}	48	Packets Transmitted [512–1023 Bytes] Count
GLPRT_PTC1522{H,L}	48	Packets Transmitted [1024 to 1522] Count
GLPRT_PTC9522{H,L}	48	Packets Transmitted [1523 to Max Bytes] Count
GLPRT_LXONRXC	32	Link XON Received Count
GLPRT_LXOFFRXC	32	Link XOFF Received Count
GLPRT_LXONTXC	32	Link XON Transmitted Count
GLPRT_LXOFFTXC	32	Link XOFF Transmitted Count
GLPRT_PXONRXC[n]	32	Priority XON Received Count
GLPRT_PXOFFRXC[n]	32	Priority XOFF Received Count
GLPRT_PXONTXC[n]	32	Priority XON Transmitted Count
GLPRT_PXOFFTXC[n]	32	Priority XOFF Transmitted Count
GLPRT_CRCERRS	32	CRC Error Count
GLPRT_ILLERRC	32	Illegal Byte Error Count

**Table 7-225. Additional Per-Port Counters (Continued)**

Register Names	Width	Description
GLPRT_MLFC	32	MAC Local Fault Count
GLPRT_MRFC	32	MAC Remote Fault Count
GLPRT_RLEC	32	Receive Length Error Count
GLPRT_RUC	32	Receive Undersize Count
GLPRT_RFC	32	Receive Fragment Count
GLPRT_ROC	32	Receive Oversize Count
GLPRT_RJC	32	Receive Jabber Count
GLPRT_RDPC	32	Received packets from the network that are dropped in the receive packet buffer. The packets are dropped due to possible lack of bandwidth on the PCIe or total bandwidth of the internal data path.
GLPRT_LDPC	32	Same as the GLPRT_RDPC for VM to VM loopback packets.
GLPRT_TDOLD	32	Transmit Packets Dropped On Link Down
GLPRT_RUPP	32	Packets received on this port that were dropped because they did not match any S-TAG. Relevant only if a PV is used on this port, should be ignored otherwise.

7.11.4.2 VEB Statistics

The X710-TM4 supports SMON statistics per RFC 2613 and smonPrioStats counters per priority (see [Table 7-226](#)) for up to 16 VEBs.

The X710-TM4 supports a set of smonVlanStats counters for up to 128 802.1Q VLANs listed in [Table 7-227](#).

The counters available per VEB/UP are listed in [Table 7-226](#):

Table 7-226. VEB per UP Counters

Register Name	Width	Description
GLVEBTC_RPC{H,L}	48	Total number of packets received per priority
GLVEBTC_RBC{H,L}	48	Total number of octets received per priority
GLVEBTC_TPC{H,L}	48	Total number of packets transmitted per priority
GLVEBTC_TBC{H,L}	48	Total number of octets transmitted per priority

Table 7-227. VEB per VLAN Counters

Register Name	Width	Description
GLVEBVL_GORC{H,L}	48	Incoming packet octets
GLVEBVL_GOTC{H,L}	48	Outgoing packet octets
GLVEBVL_UPC{H,L}	48	number of unicast packets
GLVEBVL_MPC{H,L}	48	number of multicast packets
GLVEBVL_BPC{H,L}	48	number of broadcast packets



Table 7-228. Per VSI Counters

Register Name	Width	Description
GLV_TEPC	32	Transmit error packet count.
GLV_GOTC{H,L}	48	Transmit octet count. Counts number of bytes transmitted by this VSI.
GLV_GORC{H,L}	48	Receive octet count. Counts number of bytes received by this VSI. Will count packets forwarded by switch even if dropped by later stages like flow director.
GLV_UPRC{H,L}	48	Receive unicast packet count. Counts number of unicast packets received by this VSI. Will count packets forwarded by switch even if dropped by later stages like flow director.
GLV_RUPP{H,L}	48	Receive Packets dropped because of an unknown protocol or no forward destination.
GLV_BPTC{H,L}	48	Transmit Broadcast packet count. Counts number of broadcast packets transmitted by this VSI.
GLV_MPTC{H,L}	48	Transmit multicast packet count. Counts number of multicast packets transmitted by this VSI.
GLV_MPRC{H,L}	48	Receive multicast packet count. Counts number of multicast packets received by this VSI. Will count packets forwarded by switch even if dropped by later stages like flow director.
GLV_BPRC{H,L}	48	Receive Broadcast packet count. Counts number of broadcast packets received by this VSI. Will count packets forwarded by switch even if dropped by later stages like flow director.
GLV_UPTC{H,L}	48	Transmit unicast packet count. Counts number of unicast packets transmitted by this VSI.
GLV_RDPC	32	Counts (per VSI) packets that were drop due to no descriptors in host queue.

The X710-TM4 provides virtual bridge port or interface statistics counters for VSIs and uplinks associated with a VEB instance. See section 7.11.5 for additional details on VSI and uplink interface statistics.

Note: Port/VSI based mirror packets are not counted in the VEB, VEB/TC counters.

Transmits counters usually counts only packet that are sent either to the LAN or to the loopback. In some cases, it may count packets that are dropped later when the packet is sent back to the receive. For example, a valid transmit packet dropped in receive due to VSI source pruning is counted in Tx counters.

Note: Do not count packets not sent to the LAN.

7.11.4.3 Statistics Resources

Almost all statistic counters are statically allocated. This means that for each element type there is a 1:1 ratio between the number of elements and the number of counter sets. The only two exceptions to this are VEB per VLAN statistics. There can be up to 256 VEBs (switch IDs), but only 32 of these may be allocated a statistic block. The total number of VLAN x VEB combinations in the system can exceed the number of counter sets.

For statically allocated counters, the admin command allocating the object will return the offset in the counter array of the counters for this entity. For dynamically allocated counters, the allocator must request statistics counters for the object. If stat counters are not available, the allocation will fail. The allocating driver may retry the operation without requesting stats in that case.

7.11.5 Other Statistics

This section includes statistic counters that are not included in the previous sections.



Table 7-229. Other Statistics

Name	Description	Size	Comment
GL_RXERR1 {H,L}	Receive Error Counter 1	64 bits	Count dropped packet due to one of the following exceptions: <ul style="list-style-type: none"> - Packet size is larger than RXMAX of the queue - Internal Receive queue context error (could be a result of a reset in progress) - Receive descriptor Unsupported Request on the PCI or internal Dummy completion (the Dummy completion is a result of a reset in progress)
GL_RXERR2 {H,L}	Receive Error Counter 2	64 bits	Count dropped packet due to one of the following exceptions: <ul style="list-style-type: none"> - Packets directed to invalid receive queues - Packets directed to disabled receive queues - Packets dropped by the switch filters (these packets are counted also by the switch statistics) - Packets dropped due to MAC errors or FC CRC errors - Packets dropped by the FD filter - Packets dropped due to VM reset, VF reset or PF reset



7.12 LLDP Protocol

7.12.1 Introduction

The X710-TM4 supports the IEEE Data Center Bridging standards such as IEEE 802.1Qaz - Enhanced Transmission Selection (ETS), IEEE 802.1Qbb - Priority based Flow Control (PFC), IEEE 802.1Qbg - Edge Virtual Bridging (ECB) and IEEE Std. 802.3az - Energy Efficient Ethernet (EEE). Devices that support these standards use IEEE 802.1AB Link Layer Discovery Protocol (LLDP) to exchange configuration information with their network link partner.

LLDP is a link layer protocol that enables a LAN station to advertise capabilities and status of the system. An LLDP agent transmits and receives information to and from the LLDP agents of other stations attached to the same LAN. The information distributed and received in each LLDPDU (LLDP Data Unit) is stored in two MIBs (Management Information Base) per physical LAN port, one for the nearest bridge and the other for non-TPMR.

7.12.2 Scope

The X710-TM4 supports an embedded LLDP agent that runs on the Embedded Management Processor (EMP).

This section describes the implementation of the embedded LLDP agent. It also describes the agents operational modes, supported TLVs, configuration and run time operation of LLDP.

7.12.3 LLDP Agent

The following IEEE standards have configuration parameters and use LLDP to exchange configuration parameters with the link partner.

- Edge Virtual Bridging (EVB) per IEEE 802.1Qbg
- Data Center Bridging (DCB)
- Energy Efficient Ethernet (EEE)

The IEEE 802.1Qbg specification enables a service provider to support multipoint LAN services to customers over a single physical link by using multiple virtual channels, known as S-Channels. The initial enablement and setting of these capabilities is done via a CDCP TLV.

The DCB features (PFC, ETS, DCBX, App TLV) are defined as requirements for a VLAN aware bridge component. Both C-components and S-components are VLAN-aware bridge components. The X710-TM4 supports a single instance of DCB and EEE logic per physical LAN port. Therefore, the X710-TM4 DCB logic is associated with the component connected to the physical LAN port, either C or S depending on the use case/configuration.

The X710-TM4 supports an LLDP agent that exchanges LLDP MIB with its peer. The LLDP agent exchanges the LLDP MIB with either an S-VLAN bridge or C-VLAN bridge depending on mode of operation.

The LLDP agent is active under the following conditions:

1. During pre-boot operations including S5 (D0u and D0a)



2. During operating system present mode unless software explicitly turns off the agent using the Stop LLDP Agent AQ command.

During DCBX exchange, the LLDP agent sets the willingness bit and accepts a recommended setting from its link partner.

The LLDP agent is enabled during power-on by setting the NVM *LLDP Admin Status* field, (enabled separately per Ethernet port). It is disabled when the Stop LLDP Agent is executed (per Ethernet port). Software might transfer ownership of LLDP processing back to the device by issuing a Start LLDP Agent AQ command.

7.12.4 LLDP Processing

7.12.4.1 LLDPDU Addressing and Forwarding

7.12.4.1.1 Egress Rules

On the egress side, the LLDP agent uses the appropriate group MAC address as destination MAC and with Ether type 0x88CC:

- DCB and EEE - Nearest bridge address
- CDCP - Non-TPMR nearest bridge address

When an LLDP agent is enabled in the device, the following forwarding rules apply for untagged LLDP packet originating in the host:

- Packets with a nearest bridge destination MAC address are forwarded to the internal control port.
- Packets with the nearest customer bridge destination MAC address are dropped.
- Packets with a non-TPMR destination MAC address are forwarded to the internal control port.

A control port that needs to send control packets overriding these rules should use the *SWTCH* field in the transmit context descriptor of the control packet. Tagged (802.1Qbg) LLDP packets are forwarded like regular packets. The software device driver might define different rules (forward to control VSI or drop) using the Add Control Packet Filter admin command ([Section 7.4.9.5.9.9](#)).

7.12.4.1.2 Ingress Rules

When an LLDP agent is enabled in the device, the following forwarding rules apply for RX LLDP packets:

Table 7-230. Forwarding Rules for Rx LLDP Packets

Destination MAC Address	S-tagged LLDP Packets	Untagged LLDP Packets
Nearest Bridge Address	Forward to PF if programmed so by the PF. Drop otherwise	Forward to EMP
Non-TPMR Address	Forward to PF if programmed so by the PF. Drop otherwise	Forward to EMP
Nearest Customer Bridge Address	Forward to PF if programmed so by the PF. Drop otherwise	Forward to PF if programmed so by the PF. Drop otherwise



7.12.4.2 Supported TLV

The LLDP agent should include following TLV as part of the LLDPDU.

Chassis ID TLV — Is one of 4 mandatory TLVs. Uses TLV type value of 1 and subtype value of 4 (MAC address). This TLV contains the permanent/factory assigned MAC address of the LAN port.

Port ID TLV — Is one of 4 mandatory TLVs. Uses TLV type value of 2 and subtype value of 3 (MAC address). This TLV contains MAC address of any physical function. If there is none assigned to a physical function, then this TLV contains the permanent/factory assigned MAC address of the LAN port.

TTL TLV — Is one of 4 mandatory TLVs. Uses TLV type value 3. This TLV contains time to live value and is the lower between $((\text{msgTxHold} * \text{msgTxInterval}) + 1)$ and 65535. Default msgTxHold and msgTxInterval are defined in NVM and are loaded by the agent during initialization. Please refer to LLDP Configuration section for default values.

End of LLDPDU TLV — Is one of 4 mandatory TLVs. Uses TLV type value 0. This TLV does not contain any information and sets the TLV information string length to 0. Note: Although this is a mandatory TLV, there are apparently some implementations out there which do not send it. The device therefore does not require that an LLDPDU ends with this TLV.

OEM Device type TLV — Is one of the 3 OEM required TLVs. This TLV uses TLV type value of 127 with subtype of 1. Please refer to OEM specific LLDP specifications for OUI and contents of the TLV.

OEM Firmware Version TLV — Is one of the 3 OEM required TLVs. This TLV uses TLV type value of 127 with subtype of 3. Please refer to OEM specific LLDP specifications for OUI and contents of the TLV.

OEM Port Capabilities TLV — Is one of the 3 OEM required TLVs. This TLV uses TLV type value of 127 with subtype of 4. Please refer to OEM specific LLDP specifications for OUI and contents of the TLV.

DCBX ETS Configuration TLV — This TLV uses type value of 127 with OUI of IEEE 802.1 which is 0x0080C2. Subtype is 0x09.

This TLV information string uses the following default values:

- Willing bit set to 1 indicating that this station is willing to accept configuration from remote station.
- CBS bit is set to 0 indicating that this station does not support credit based shaper.
- Maximum TCs
- Priority assignment table is a 4 octet string with 4 bits per entry.
- Bandwidth table is a string of 8 octet string with each octet defining the bandwidth percentage for traffic classes. Octet 0 specifies bandwidth percentage of Traffic class 0, octet 1 for traffic class 1 and so on.
- TSA Assignment table is a string of 8 octets with 8 bit value specifying Transmission selection algorithm per traffic class.

DCBX PFC Configuration TLV — This TLV uses type value of 127 with OUI of IEEE 802.1 which is 0x0080C2. Subtype is 0x0B.

This TLV information string uses the following default values:

- Willing bit set to 1 indicating that this station is willing to accept configuration from remote station.
- PFC Capability is a 4 bit unsigned integer indicating the number of traffic classes that simultaneously support PFC. Default value is 8.
- PFC Enable is a 8 bit vector that indicates which traffic classes have PFC enabled. Default value is zero. None of the traffic call uses have PFC enabled by default.

DCBX Application Priority Configuration TLV — This TLV uses type value of 127 with OUI of IEEE 802.1 which is 0x0080C2. Subtype is 0x0C.



By default this optional TLV is not transmitted by the LLDP agent but the agent reflects the table received from peer network port if the agent receives LLDPDU with remote link partner.

EEE TLV — EEE TLV is included in the LLDPDU only if both link partners advertise EEE capability for resolved PHY type during auto negotiation.

This TLV uses type value of 127 with OUI of IEEE Std. 802.3-2018 which is 00-12-0F. Subtype is 5. EEE TLV information string is 10 octets long. TLV information string carries timing information following Low Power Idle (LPI) exit.

First six octets contain TransmitTw, ReceiveTw and FallbackTw. Default values for TransmitTw and Receive Tw values are loaded from NVM. Value of FallbackTw is same as ReceiveTw (FallbackTw = ReceiveTw). TransmitTw and ReceiveTw can be changed at runtime by using admin queue commands from a physical function.

Note: There is a single LLDP agent for all LAN ports and each LAN port has multiple Physical functions. Therefore, it is likely that these variables can be overwritten by multiple admin queue commands. Last admin command wins.

Echo Transmit Tw and Echo Receive Tw values are of the remote link partner. If the link partner has not transmitted an LLDPDU then HW default value for the resolved PHY type.

CDCP TLV — This TLV uses type value of 127 with OUI of IEEE 802.1 which is 0x0080C2. Subtype is 0x0E

The TLV information string uses the following values:

- Role - indicates if sender is station or bridge. Set to 1b for station
- SComp - indicates the presence or absence of an S-VLAN component for S-channel support. Set according to the NVM EVB protocols *Enabled* field
- ChnCap - identifies the total number of S-channels, both assigned and available to be assigned, that the sender has.
- (SCID/S-VID) list - An SCID/S-VID pair exists for each S-channel that is currently supported by the sender

SCID - indicates the index number of the S-channel. Zero indicates reserved space in the TLV

S-VID - The VLAN ID assigned to the S-channel

For the link speeds 2.5 Gb/s and 5 Gb/s all TLV's listed in [Table 7-231](#) are unsupported and should never be processed by firmware (both IEEE and CEE).

Table 7-231. DCBx TLVs List

IEEE TLVs	CEE TLVs
ETS Configuration TLV	Priority Groups TLV
ETS Recommendation TLV	Control TLV
PFC Configuration TLV	Priority Flow Control TLV
Application Priority Configuration TLV	Priority Groups TLV



7.12.4.3 LLDP Admin Commands

7.12.4.4 LLDPDU Transmission and Reception

First LLDP transmission is immediate after the LLDP agent reaches ready state and link is up. This first transmission populates the LLDPDU with default TLV values.

The LLDP agent enters Fast transmission state/period whenever the LLDP agent detects a new neighbor. The LLDP agent enters new neighbor detected state (fast transmission period) when an IEEE 802.1az Link Layer LLDP frame with new chassis and port ID is received. Fast transmission state exited after transmitting txFastInit number of LLDPDUs. Default value of txFastInit is 4.

When not in fast transmission mode, LLDP agent transmits every msgTxInterval unless there is change in local LLDP MIB variables. LLDPDU is transmitted immediately, without waiting for msgTxInterval when there is a change in local MIB variables.

The X710-TM4 supports a single LLDP neighbor per supported destination LLDP address per port. (like two total neighbors – one for nearest bridge and one for nearest non-TPMR per port). LLDPUs with other addresses are ignored.

The transmission and reception flow is interrupted by several events, each causing a disruption in the normal flow. The behavior in such cases is as follows:

- LLDP events visible to both ends
 - Aging of the LLDP timer or LLDPDU is received with a TTL value of zero (such as shutdown LLDPDU)
 - Delete all information in the LLDP remote systems MIB associated with the respective MSAP identifier
 - EMP reconfigures device based on the local defaultsException: See Flow Control configuration
 - Missing TLV in a received LLDPDU
 - Delete all information for the missing TLV in the LLDP remote systems MIB associated with the respective MSAP identifier
 - EMP reconfigures device based on the local defaults of this TLVException: See flow control configuration
- Events that cause a link down
 - Link down event (only if LLDP timer has not expired)
 - When link is back up, delete all information in the remote systems MIB associated with the LLDP agent(s) for this link
 - EMP reconfigures device based on the local defaults
 - Device resets that cause Link Down (such as EMP reset, global reset, PCI resets that cause link down)
 - Device performs an internal GLOBR that brings the link down momentarily
 - Continue as in link down case
 - LLDP ownership transfer from software to firmware
 - Software issues a global reset to the device
 - Software sends a Start LLDP Agent AQ command to start the LLDP agent in firmware
 - Continue as in link down case
- Events not visible to the other end



- LLDP ownership transfer from firmware to software
 - LLDP agent terminated in device
 - No automatic change in device configuration
 - EMP waits for software commands
- Device resets that do not cause link down (such as core reset, PCI resets that do not cause link down and function-level resets)
 - Device performs an internal CORER
 - EMP reconfigures core based on the LLDP MIBs

Note: No change in MAC/PHY state including flow control configuration and operation

LLDP reception and transmission continues once the previous configuration completes. Configuration is estimated to take milliseconds, well below the LLDP timeout values (usually in seconds).

7.12.4.5 LLDP Protocol Variables

Unless otherwise specified, the LLDP agent operational state variables are set to values recommended by Section 9.2.2 of the IEEE 802. AB - 2009 standard. However, certain values can be configured via the NVM and are loaded from the NVM when the LLDP agent reaches an operational state. This occurs after a power up or a device reset. Refer to [Section 7.12.5.2](#) for LLDP protocol variables that can be configured via the NVM.

7.12.4.6 LLDP Data Store

The LLDP agent maintains sufficient information to enable a host software based management agent to support basic LLDP MIB and organizationally specific LLDP MIB extensions.

The following basic variables are maintained by the LLDP agent:

msgTxInterval - This variable defines the time interval in timer ticks between transmissions during normal transmission periods.

msgTxHold - This variable is used, as a multiplier of msgTxInterval, to determine the value of txTTL that is carried in LLDP frames transmitted by the LLDP agent.

reinitDelay - This parameter indicates the amount of delay from when adminStatus becomes disabled until re-initialization is attempted.

txCreditMax - Maximum number of consecutive LLDPDUs that can be transmitted at any time.

msgFastTx - This variable defines the time interval in timer ticks between transmissions during fast transmission periods.

txFastInit - This value determines the number of LLDPDUs that are transmitted during a fast transmission period.

adminStatus - This variable indicates whether or not the LLDP agent is enabled. The defined values for this variable are as follows:

- Integer value of 3 means the LLDP agent is enabled for reception and transmission of LLDPDUs.
- Integer value of 2 means the LLDP agent is enabled for transmission of LLDPDUs only.
- Integer value of 1 means the LLDP agent is enabled for reception of LLDPDUs only.
- Integer value of 0 means the LLDP agent is disabled for both reception and transmission.



The following statistics are maintained by LLDP agent per port:

RemTableLastChangeTime, RemTableInserts, RemTableDeletes, RemTbleDrops, RemTableAgeouts, TxPortFramesTotal, RxPortFramesDiscarded, RxPortFrameErrors, RxPortFramesTotal, RxPortTLVsDiscardedTotal, RxPortTLVsUnrecognizedTotal, RemTooManyNeighbors.

The LLDP agent maintains last received LLDPDU per port. Upper bound for size of LLDPDU is 1500 bytes. Details of data structure used for information store is implementation specific and beyond scope of this document.

The X710-TM4 provides LSAP services to the operating provided MSAP services that are used by LLDP agents hosted by system processors

7.12.5 Initialization and configuration

7.12.5.1 Initialization

As part of EMP initialization, an LLDP agent is loaded and initialized by the EMP. The LLDP agent loads default values for TLVs from the NVM. LLDP agent transitions to ready state and waits for LAN link up before transmitting the first LLDPDU.

The DCBX agent operates in slave mode and sets the willingness bit and accepts recommendations from the link partner.

7.12.5.2 LLDP Configuration

7.12.5.2.1 LLDP Protocol Variables

The following LLDP timers can be configured via NVM:

msgFastTx — This variable defines the time interval in timer ticks between transmissions during fast transmission periods. The default value of msgFastTx is 1. This value can be changed by management to any value in the range 1 through 3600.

msgTxInterval — variable defines the time interval in timer ticks between transmissions during normal transmission. The default value for msgTxInterval is 30 seconds. This value can be changed by management to any value in the range 1 through 3600.

msgTxHold — variable is used, as a multiplier of msgTxInterval, to determine the value of txTTL that is carried in LLDP frames transmitted by the LLDP agent. The recommended default value of msgTxHold is 4. This value can be changed by management to any value in the range 1 through 100.

txCreditMax — value determines maximum number of LLDPDUs that can be sent per second. This value defaults to 5. This value can be changed by management to any value in the range 1 through 10.

txFastInit — value determines the number of LLDPDUs that are transmitted during a fast transmission period. The default value of txFastInit is 4. This value can be changed by management to any value in the range 1 through 8.

reinitDelay — This parameter indicates the amount of delay from when adminStatus becomes disabled until re-initialization is attempted. Default value is 2.



7.12.5.2.2 EEE TLV Variables

The following EEE timers can be configured via admin queue commands and defaults are stored and loaded from NVM:

EEE TransmitTw — value determines the time (expressed in microseconds) that the transmitting link partner will wait before it starts transmitting data after leaving the Low Power Idle (LPI) mode. This value is platform dependent and depends on the OEM platform. NVM has a default value that is loaded by the LLDP agent. However, this variable can be configured via admin queue interface.

EEE ReceiveTw — value determines the time (expressed in microseconds) that the receiving link partner is requesting the transmitting link partner to wait before starting the transmission data following the LPI. This value is platform dependent and depends on the OEM platform. NVM has a default value that is loaded by the LLDP agent. However, this variable can be configured via admin queue interface.

7.12.5.2.3 LLDP Admin Queue Commands

The following commands are supported by X710-TM4 to manage the LLDP agent and provide information to the drivers.

Table 7-232. LLDP Admin Queue Commands

Command	OpCode	Description
Get LLDP MIB	0x0A00	Fetch latest LLDP MIB.
Configure LLDP MIB Change Event	0x0A01	Request and deliver a notification that the peer has sent an updated LLDP MIB.
Add LLDP TLV	0x0A02	Add a new TLV to the local LLDP MIB.
Update LLDP TLV	0x0A03	Update an existing TLV in the local LLDP MIB.
Delete LLDP TLV	0x0A04	Delete a TLV from the local LLDP MIB.
Stop LLDP Agent	0x0A05	Used to stop or shutdown LLDP agent.
Start LLDP Agent	0x0A06	Start an LLDP agent running
Get CEE DCBX CFG	0x0A07	Retrieves the CEE configuration
Set Local LLDP MIB	0x0A08	Load the DCBX configuration
Stop/Start a Specific LLDP Agent	0x0A09	Stop and restart the FW DCBX agent

7.12.5.2.3.1 Get LLDP MIB

This command, posted on the ATQ, is an indirect AQ command. The software device driver requests the complete LLDP MIB, providing a response buffer (address/length pair) and the type of LLDP MIB requested. The LLDP MIB is associated with a physical LAN port. Instead of formatting the LLDP MIB in any particular way, firmware should write the entire packet (including headers) that was sent/received on the wire for the particular MIB type specified. The MIB is guaranteed to fit in a 1.5K packet, so there is no need to use a large buffer. For a particular bridge type, software might request the local MIB, the remote MIB or both MIBs.

Firmware writes back the complete LLDP MIB to the response buffer. It writes the length of the LLDP MIB into the *Datalen* field in the descriptor on the ATQ. It writes the status of the request in the *Return Value* field. For the case where both MIBs are returned, firmware always writes the Local MIB first and the remote MIB immediately following the Local MIB.



For the link speeds 2.5 and 5 Gb/s, this command should support local MIB only (byte 16, bits 0 and 1 should be zero).

EMP firmware should return a default MIB configuration.

If any other MIB type is requested by software, firmware should return an *I40E_AQ_RC_EMODE* error.

Table 7-233. Get LLDP MIB Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A00	Command opcode
Datalen	4-5		Length of response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0-1: Direction. 0 = Local MIB (sent by device). 1 = Remote MIB (received by device). 2 = Both local and remote MIBs. 3 = Reserved. Bits 2-3: Bridge type. 0 = Nearest bridge. 1 = Non-TPMR bridge. 2 = Reserved. 3 = Reserved. Bits 4-7: Reserved.
Reserved	17-23	Reserved	Reserved.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		Address of response buffer.

Table 7-234. Get LLDP MIB Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A00	Command opcode.
Datalen	4-5		Length of LLDP MIB if status = success. In bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. A value of SUCCESS means that command was performed successfully. Error Codes: ENOENT - Firmware returns this value if any requested LLDP MIB doesn't exist. EPERM - Firmware returns this value if software has taken control of LLDP processing. EFBIG - Firmware returns this value when the size of LLDPDU is larger than the size of the response buffer. EINVAL - Firmware returns this value when software requests a bad request (such as invalid bridge type or direction).

**Table 7-234. Get LLDP MIB Response**

Name	Bytes.Bits	Value	Remarks
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0-1: Direction. 0 = Local MIB (sent by device). 1 = Remote MIB (received by device). 2 = Both local and remote MIBs. 3 = Reserved. Bits 2-3: Bridge type. 0 = Nearest bridge. 1 = Non-TPMR bridge. 2 = Reserved. 3 = Reserved. Bits 4-7: Reserved.
Reserved	17	Reserved	Reserved
Local MIB Length	18-19	Len	If the response buffer contains a local MIB, this field reports its length. If no local MIB is present, this field is written with a value of 0.
Remote MIB Length	20-21	Len	If the response buffer contains a remote MIB, this field reports its length. If no remote MIB is present, this field is written with a value of 0.
Reserved	22-23	Reserved	Reserved.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		Address of response buffer.

Note: If only one MIB is present, it is written to the response buffer beginning at the first byte of the response buffer (such as offset = 0). If two MIBs are present, the local MIB is always written first (offset = 0) and the remote MIB is written immediately following the local MIB (offset=LocalMIBLength).

7.12.5.2.3.2 Configure LLDP MIB Change Event

This command, posted on the ATQ, is a direct AQ command. The software device driver uses this command to request that firmware post an event on the ARQ when the LLDP MIB associated with this interface changes. The software device driver can also use this command to request that firmware stop posting this event to the ARQ.

Firmware writes back the status of the request to the *Return Value* field.

Table 7-235. Configure LLDP MIB Change Event Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A01	Command opcode.
Datalen	4-5	0	Direct command. No response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.

**Table 7-235. Configure LLDP MIB Change Event Command**

Name	Bytes.Bits	Value	Remarks
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Command	16	Cmd	Bit 0: Command. 0 = Enable event. 1 = Disable event. Bits 1-7: Reserved.
Reserved	17-31	Reserved	Reserved.

Table 7-236. Configure LLDP MIB Change Event Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A01	Command opcode.
Datalen	4-5	0	Direct command. No response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. A value of SUCCESS means that command was performed successfully. Error codes: EPRM - Firmware returns this value if software has taken control of LLDP processing.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31	Reserved	Reserved.

7.12.5.2.3.3 LLDP MIB Change Event

This event is posted on the ARQ to indicate to software that any LLDP MIB associated with the physical interface has changed. The LLDP MIB change event must also be posted if a multiple peers condition is detected or if a TLV is aged out.

Firmware provides the status of the event, the length of the LLDP MIB in bytes, the type of LLDP MIB that has changed and copies the cookie value from the associated Configure LLDP MIB Change Event.

The response buffer includes the entire MIB that changed. The formatting is identical to the response for the Get LLDP MIB command.

Note that the OpCode for this event is the same as the OpCode for the related command on the ATQ, which enabled this event to be sent to software.

Table 7-237. LLDP MIB Change Event

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A01	Event code.
Datalen	4-5		Length of LLDP MIB if Status = success. In bytes.



Table 7-237. LLDP MIB Change Event

Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0-1: Direction. 0 = Local MIB (sent by device). 1 = Remote MIB (received by device). 2 = Reserved. 3 = Reserved. Bits 2-3: Bridge type. 0 = Nearest bridge. 1 = Non-TPMR bridge. 2 = Reserved. 3 = Reserved. Bits 4-5: Miscellaneous. 0 = Port's Tx active. 1 = Port's Tx suspended and drained. 2 = Reserved. 3 = Port's Tx suspended and drained. Blocked TC pipe flushed. Bits 6-7: Reserved.
Reserved	17-23	Reserved	Reserved.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		Address of response buffer.

7.12.5.2.3.4 Add LLDP TLV

This command, posted on the ATQ, is an indirect AQ command. The software device driver provides the type of MIB to be updated, the TLV to be added and the address/length of the buffer containing the TLV. Software is responsible for guaranteeing that the TLV to be added does not already exist in the MIB or follows the TLV usage rules for TLVs that enable multiple instances. Software can only add TLVs to the Local LLDP MIB.

Firmware adds the new TLV to the Local LLDP MIB just before the End of LLDPDU TLV. Firmware writes back the complete LLDP MIB to the response buffer. It writes the length of the LLDP MIB into the *Datalen* field in the descriptor on the ATQ.

For the link speeds 2.5 Gb/s and 5 Gb/s all TLV's listed [Table 7-231](#) are unsupported and should never be processed by firmware (both IEEE and CEE).

EMP firmware should check the TLV type upon receiving the AQC.

If an unsupported type, EMP firmware should send an *I40E_AQ_RC_EMODE* response.

Table 7-238. Add LLDP TLV Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section Section 7.10.5.2.1 for details.
Opcode	2-3	0x0A02	Command opcode.
Datalen	4-5	Len	Length of indirect buffer.



Table 7-238. Add LLDP TLV Command

Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Return value. Zeroed by software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0-1: Reserved Bits 2-3: Bridge type. 0=Nearest bridge. 1=Non-TPMR bridge. 2=Reserved. 3=Reserved. Bits 4-7: Reserved.
Reserved	17	Reserved	Reserved.
Length	18-19	Len	Length of TLV placed in the indirect buffer by the driver.
Reserved	20-23	Reserved	Reserved.
Data Address High	24-27		Address of indirect buffer.
Data Address Low	28-31		Address of indirect buffer.

Table 7-239. Add LLDP TLV Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0A02	Command opcode.
Datalen	4-5		Length of LLDP MIB if status = success. In bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. Error codes: ENOMEM – Firmware returns this value if there is not enough space available to add the new TLV. EINVAL – Firmware returns this value if the MIB type associated with this command does not exist. EPERM - Firmware returns this value if software has taken control of LLDP processing.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0-1: Reserved. Bits 2-3: Bridge type. 0 = Nearest bridge. 1 = Non-TPMR bridge. 2 = Reserved. 3 = Reserved. Bits 4-7: Reserved.

**Table 7-239. Add LLDP TLV Response**

Name	Bytes.Bits	Value	Remarks
Reserved	17-23	Reserved	Reserved.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		Address of response buffer.

7.12.5.2.3.5 Update LLDP TLV

This command, posted on the ATQ, is an indirect AQ command. The software device driver provides the bridge type of the MIB to be updated, the TLV to be updated (both original and updated versions), the offset/length of both TLVs in the indirect buffer and the address/length of the indirect buffer. Software is responsible for guaranteeing that the TLV to be updated does already exist in the MIB. Only a local MIB might be updated by the software device driver.

Firmware must match the entire original TLV in the MIB before performing an update. There are some TLV types that might appear more than once and so matching based only on type/subtype is not sufficient.

Firmware writes back the complete LLDP MIB to the response buffer. It writes the length of the LLDP MIB into the *Datalen* field in the descriptor on the ATQ.

For the link speeds 2.5 Gb/s and 5 Gb/s all TLV's listed [Table 7-231](#) are unsupported and should never be processed by firmware (both IEEE and CEE).

EMP firmware should check the TLV type upon receiving the AQC.

If an unsupported type, EMP firmware should send an *I40E_AQ_RC_EMODE* response.

Table 7-240. Update LLDP TLV Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0A03	Command opcode.
Datalen	4-5	Len	Length of indirect buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0-1: Reserved. Bits 2-3: Bridge type. 0 = Nearest bridge. 1 = Non-TPMR bridge. 2 = Reserved. 3 = Reserved. Bits 4-7: Reserved.
Reserved	17	Reserved	Reserved
Length1	18-19	Len	Length of original TLV in the indirect buffer. Offset is assumed to be zero.
Offset2	20-21	Offset	Offset of updated TLV in the indirect buffer.



Table 7-240. Update LLDP TLV Command

Name	Bytes.Bits	Value	Remarks
Length2	22-23	Len	Length of updated TLV in the indirect buffer.
Data Address High	24-27		Address of indirect buffer.
Data Address Low	28-31		Address of indirect buffer.

Table 7-241. Update LLDP TLV Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0A03	Command opcode.
Datalen	4-5		Length of LLDP MIB if status - success. In bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. Error codes: EINVAL – Firmware returns this value if the requested MIB does not exist. ENOXIO – Firmware returns this value if the original TLV does not exist in the requested MIB. ENOMEM – Firmware returns this value if the updated TLV is larger than the original TLV and there is not enough space to increase the size of the TLV. EPERM - Firmware returns this value if software has taken control of LLDP processing
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0-1: Reserved. Bits 2-3: Bridge type. 0=Nearest bridge. 1=Non-TPMR bridge. 2=Reserved. 3=Reserved. Bits 4-7: Reserved.
Reserved	17-23	Reserved	Reserved.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		Address of response buffer.

7.12.5.2.3.6 Delete LLDP TLV

This command, posted on the ATQ, is an indirect AQ command. The software device driver provides the type of MIB to be updated and a copy of the TLV to be deleted. Software is responsible for guaranteeing that the TLV to be deleted does already exist in the MIB.

Firmware writes back the complete LLDP MIB to the response buffer. It writes the length of the LLDP MIB into the *Datalen* field in the descriptor on the ATQ.

For the link speeds 2.5 Gb/s and 5 Gb/s all TLV's listed [Table 7-231](#) are unsupported and should never be processed by firmware (both IEEE and CEE).

EMP firmware should check the TLV type upon receiving the AQC.



If an unsupported type, EMP firmware should send an *I40E_AQ_RC_EMODE* response.

Table 7-242. Delete LLDP TLV Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0A04	Command opcode.
Datalen	4-5	Len	Length of indirect buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0-1: Reserved. Bits 2-3: Bridge type. 0 = Nearest bridge. 1 = Non-TPMR bridge. 2 = Reserved. 3 = Reserved. Bits 4-7: Reserved.
Reserved	17	Reserved	Reserved
Length	18-19	Len	Length of TLV to be deleted. The TLV itself is copied into the indirect buffer by the software device driver.
Reserved	20-23	Reserved	Reserved.
Data Address High	24-27		Address of indirect buffer.
Data Address Low	28-31		Address of indirect buffer.

Table 7-243. Delete LLDP TLV Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0A04	Command opcode.
Datalen	4-5		Length of LLDP MIB if status = success. In bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. Error codes: EINVAL – Firmware returns this value if the requested MIB does not exist. ENOXIO – Firmware returns this value if the TLV to be deleted does not exist in the requested MIB. EPERM - Firmware returns this value if software has taken control of LLDP processing.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.



Table 7-243. Delete LLDP TLV Response

Name	Bytes.Bits	Value	Remarks
Type	16	MIB Type	Bit 0-1: Reserved. Bits 2-3: Bridge type. 0 = Nearest bridge. 1 = Non-TPMR bridge. 2 = Reserved. 3 = Reserved. Bits 4-7: Reserved.
Reserved	17-23	Reserved	Reserved.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		Address of response buffer.

7.12.5.2.3.7 Stop LLDP Agent

This command, posted on the ATQ, is a direct AQ command. The software device driver uses this command to request that firmware stop or shutdown the LLDP agent on the port.

If stop is specified, the device stops the LLDP agent on the port and directs all untagged ingress LLDP frames received on the port to the default queue of the control VSI associated with the port aggregator or port extender.

If shutdown is specified, the device stops the LLDP agent on the port and sends a last LLDP PDU on the wire with TTL = 0 and with nearest bridge and non-TPMR destination address. Firmware then directs all untagged ingress LLDP frames received on the port to the default queue of the S-component control VSI.

Firmware writes back the status of the request.

Any preceding registration to events on the port (via the Configure LLDP MIB Change Event command) is discarded. Software should register for events again once LLDP agent in the device is active again.

After the response, the software device driver should route the LLDP flows to a control VSI using the Add Control Packet Filter command ([Section 7.4.9.5.9.9](#)).

Table 7-244. Stop LLDP Agent Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A05	Command opcode.
Datalen	4-5	0	Direct command. No response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by Firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Command	16	Cmd	Bit 0: Command. 0 = Stop LLDP agent. 1 = Shutdown LLDP agent. Bits 1-7: Reserved.
Reserved	17-31	Reserved	Reserved.



Table 7-245. Stop LLDP Agent Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A05	Command opcode.
Datalen	4-5	0	Direct command; no response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. A value of SUCCESS means that the command was performed successfully. Error codes: EPRM - Firmware returns this value if software has taken control of LLDP processing.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31	Reserved	Reserved.

7.12.5.2.3.8 Start LLDP Agent

It is expected that CORER has occurred before this command is issued. CORER cause sthe EMP to reload LLDP forwarding rules from NVM and re-initialize per NVM settings.

This command, posted on the ATQ, is a direct AQ command. In response to this command EMP re-enables LLDP agent over all ports enabled by the NVM LLDP Admin Status word. It is expected that only pre-boot software issues this command to start the LLDP agent.

If the software device driver defined LLDP forwarding rules previous to sending this command, these rules should be removed using the Remove Control Packet Filter admin command (Section 7.4.9.5.9.10) before sending this command.

Table 7-246. Start LLDP Agent Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A06	Command opcode.
Datalen	4-5	0	Direct command; no response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Command	16	Cmd	Bit 0: Command. 0 = Do not start the LLDP agent. Note that this value should not be used). 1= Start LLDP agent. Bits 1-7: Reserved.
Reserved	17-31	Reserved	Reserved.

**Table 7-247. Start LLDP Agent Response**

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A06	Command opcode.
Datalen	4-5	0	Direct command; no response buffer.
Return Value/ VFID	6-7		Return value. The following error values can be returned: EEXIST - LLDP agent is already running in firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-31	Reserved	Reserved.

7.12.5.2.3.9 Get CEE DCBX OPER CFG

This command, posted on the ATQ, is an indirect AQ command. The software device driver requests the operational configuration of CEE/DCBX. The software driver provides a response buffer (address/length pair).

The EMP writes back the CEE/DCBX configuration to the response buffer.

Table 7-248. Get CEE DCBX OPER CFG Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A07	Command opcode.
Datalen	4-5		Length of response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-23	Reserved	Reserved.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		Address of response buffer.

Table 7-249. Get CEE DCBX OPER CFG Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A07	Command opcode.
Datalen	4-5		Length of LLDP MIB if status = success. In bytes.



Table 7-249. Get CEE DCBX OPER CFG Response

Name	Bytes.Bits	Value	Remarks
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. A value of SUCCESS means that command was performed successfully. Error codes: ENOENT - Firmware returns this value if any requested LLDP MIB doesn't exist. EPERM - Firmware returns this value if software has taken control of LLDP processing.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Reserved	16-23	Reserved	Reserved.
Data Address High	24-27		Address of response buffer.
Data Address Low	28-31		Address of response buffer.

Table 7-250. Get CEE DCBX OPER CFG Response Buffer Format

Offset (bytes)	Description
0	Local Oper Num Traffic Class Supported. Returns a value between 1-8 for the number of TC's supported locally.
1-4	Local Oper Priority Assignment. For each UP a 4-bits for the TCID. available value is 0-7. upper bit = 0. Bits 0-3 assigned for up 0. Bits 4-7 assigned for up 1. : Bits 27-31 assigned for up 7.
5-12	Local Oper TCB Bandwidth. An 8-byte field. Each byte represents the relative BW allocation of one enabled TC. Byte 7 assigned for TC 0. Relative BW allocation is a value 0-100 and represents the percentage of the available BW this TC is allocated with. The sum of the BW allocated for all TC equal to 100%.
13	Local Oper PFC Enable. A bitmap containing a PFC-enable flag for each UP. Bit 0 concerns UP 0.

**Table 7-250. Get CEE DCBX OPER CFG Response Buffer Format**

Offset (bytes)	Description
14-15	Local Oper Application Priority. Indicates the Application TLV negotiated for iSCSI and FIP. (encoded in three bits). 17.0-2 = 17.3-5 = Reserved for iSCSI application priority. 17.6-7 = Reserved. 18.0-2 = Reserved for FIP Application priority. 18.3-7 = Reserved
16-19	Status Flags for DCBX TLVs. For each TLV indicates the status of the TLV (1 = True, 0 = False): Bit 0 - Operational Mode ; Bit 1 - Synced ; Bit 2 - Error. 16.0-2 = Status bits for PG. 16.3-5 = Status bits for PFC. 16.6-7 = Reserved. 17.0-2 = Reserved. 17.3-5 = Status bits for iSCSI Application Priority. 17.6-7 = Reserved. 18.0-2 = Status bits for FIP Application Priority. 18.3-7 = Reserved. 19.0-7 = Reserved.
20-31	Reserved, return zero.

7.12.5.2.3.10 Set Local LLDP MIB

This command, posted on the ATQ, is an indirect AQ command. The software device driver configures the complete DCBX MIB, providing a response buffer (address/length pair). The DCBX MIB is associated with a physical LAN port and must be expressed in IEEE format even though firmware finally sends it as CEE TLVs on the wire if needed. The MIB is guaranteed to fit in a 1.5K packet, so there is no need to use a large buffer. Once DCBX negotiation with the peer completes, the local DCBX MIB returned by the Get LLDP MIB command reflects the final resolved values. In the future, the command could be extended to support more other LLDP MIBs.

This command is useful to configure the local DCBX agent into the non-willing mode such as master mode and to set the DCB configuration to be pushed to the peer.

Firmware writes back the status of the request in the *Return Value* field.

If the willing bit is set in a DCBX TLV, the DCBX agent should take it in account when resolving DCBX with the peer, as per the rules defined in the IEEE/CEE standard.

Firmware might change the local configuration twice, once upon reception of the AQ command to align default configuration to what is published in the TLVs sent to the peer and once upon reception of the peer's TLV when resolving DCBX. These two steps might be collapsed into one single configuration change in case peer's TLV is received within short delays. The new default configuration is maintained until the next GLOBR event or until a Stop DCBX Agent AQ command is received.

If the command is received while the firmware DCBX agent is disabled or stopped, the MIB is parsed by firmware and used to configure the local DCB settings of the port, with no DCB TLV exchange with the peer performed by firmware. Firmware must drain the Tx pipe if TC or PFC changes were pushed, as if it was resulting from a regular DCBX negotiation flow.

This command is not supported in 2.5 Gb/s and 5 Gb/s link speed modes.

The command is ignored by EMP firmware and an *I40E_AQ_RC_EMODE* response should be sent.



Table 7-251. Set Local LLDP MIB Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A08	Command opcode.
Datalen	4-5		Length of response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0: 0 = Local DCBX MIB (sent by device). Bit 1: 0 = CEE DCB, APP TLV operates in willing mode. 1 = CEE DCB, APP TLV operates in non willing mode. Bits 2-7: Reserved.
Reserved	17	Reserved	Reserved.
Local MIB Length	18-19	Len	Length of the command buffer.
Reserved	20-23	Reserved	Reserved.
Data Address High	24-27		Address of command buffer.
Data Address Low	28-31		Address of command buffer.

Table 7-252. Set Local LLDP MIB Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A08	Command opcode.
Datalen	4-5		Length of LLDP MIB if status = success. In bytes.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. A value of SUCCESS means that the command was performed successfully. Error codes: EPERM - If any local LLDP MIB set in the command doesn't exist (like if the specific agent was stopped). EINVAL - If a DCBx TLV is missing in the pushed MIB or if the TLVs pushed are not consistent or illegal.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.



Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Type	16	MIB Type	Bit 0: 0 = MIB is silently changed. Event to software is not required. 1 = MIB is Changed, MIB change event is triggered. Bits 1-7: Reserved
Reserved	17-31	Reserved	Reserved

7.12.5.2.3.11 Stop/Start A Specific LLDP Agent

This command, posted on the ATQ, is a direct AQ command. The software device driver uses this command to request that firmware stops or re-starts the DCBX agent on the port. In the future, the command could be extended to support more specific agents.

Stopping the DCBX agent on the port means the following:

1. When parsing the LLDP TLVs received from the peer, the LLDP agent skips over all DCBX TLVs.
2. LLDP agent does not send DCBX TLVs to the peer.
3. Local DCB configuration is returned to the hardware default (single TC, no PFC).
4. Get LLDP MIB still returns the remote DCBX MIBs if such are received from the peer.

Restarting the DCBX agent on the port means the following:

1. LLDP agent parses and processes the DCBX TLVs received from the peer.
2. LLDP agent sends DCBX TLVs to the peer.
3. Local DCB configuration is modified according to the DCBX negotiation with peer.

Firmware writes back the status of the request.

The stop only option (bit 16.0 = 0) is supported in 2.5 Gb/s and 5 Gb/s link speed modes. In all other cases the command should be ignored by EMP firmware and an *I40E_AQ_RC_EMODE* response should be sent.

Table 7-253. Stop/Start A Specific LLDP Agent Command

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A09	Command opcode.
Datalen	4-5	0	Direct command. No response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Command	16	Cmd	Bit 0: Command. 0 = Stop DCBX agent. 1 = Re-start DCBX agent. Bits 1-7: Reserved.



Reserved	17-31	Reserved	Reserved
----------	-------	----------	----------

Table 7-254. Stop/Start A Specific LLDP Agent Response

Name	Bytes.Bits	Value	Remarks
Flags	1:0		See Admin Queue section for details.
Opcode	2-3	0x0A09	Command opcode.
Datalen	4-5	0	Direct command. No response buffer.
Return Value/ VFID	6-7		Return value. Zeroed by the software device driver. Written by firmware. Status of request. A value of SUCCESS means that the command was performed successfully. Error codes: EMODE - FW returns this value in MFP modes.
Cookie High	8-11	Cookie	Opaque value. Copied by firmware into the completion of this command.
Cookie Low	12-15	Cookie	Opaque value. Copied by firmware into the completion of this command.
Status	16	Cmd	Bit 0: Status. 0=DCBX agent stopped. 1=DCBX agent active. Bits 1-7: Reserved.
Reserved	17-31	Reserved	Reserved.



NOTE: *This page intentionally left blank.*



8.0 LAN Engine

8.1 LAN Cache and Private Host Memory

The LAN transmit queue contexts are stored in memory pages in the Function Private Memory (FPM) space. The FPM for the Tx queue contexts is defined by the GLHMC_LANTXBASE and GLHMC_LANTXOBSZ registers. The context of the PF and its VFs reside in the private memory spaces of the PF.

The FPM is considered a private memory of the hardware; software is not expected to access it other than (as described in this section). During run time, the hardware fetches the queue context from the FPM to its cache according to internal needs.

8.2 Shared LAN Data Path

8.2.1 LAN Queue Pairs Allocation

8.2.1.1 LAN Queue Pairs Allocation to PFs

Queues are allocated in pairs of transmit and receive queues, called LAN queue pairs (LQP) in this section. LQPs are statically allocated to PFs in a flexible manner according to PFLAN_QALLOC registers (loaded from the NVM):

- Active PFs must have LQPs enabled by the VALID flag.
- LQPs of a PF begin at the absolute LQP index defined by the *FIRSTQ* field and end at the absolute LQP index defined by the *LASTQ* field.

The allocated LQPs for the PFs must be defined monotonically. This means that LQPs of PF(n) must be allocated after the LQPs of PF(n-1) in the absolute physical device space. See an example of 4 PFs in [Figure 8-1](#).

Note: Based on the PFLAN_QALLOC values, the PF software allocates the LQPs for its usage and for its VFs as detailed in the following subsections.

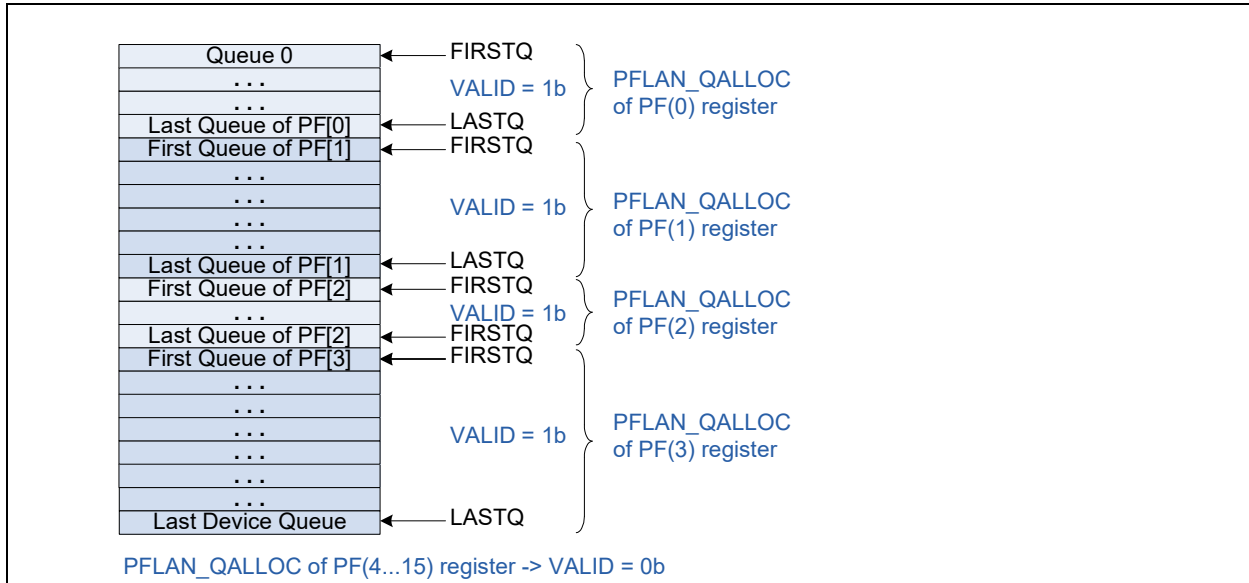


Figure 8-1. LAN Queue Pairs Allocation Example of Four PFs

8.2.1.2 LAN Queue Pairs Allocation Within a PF

The PF software device driver is expected to read the values of the *PFLAN_QALLOC* register to determine the LQPs owned. The PF allocates these LQPs by programming the *VPLAN_QTABLEs* registers and by the add VSI Admin Queue command which programs the *VSILAN_QBASE* and *VSILAN_QTABLEs*. An example of LQP allocation within a PF to its VSIs is shown in [Figure 8-2](#).

- Receive queue zero of the PF is reserved for Flow Director Filter Programming Status Descriptor (see [Section 8.3.2.2.3](#)). LPQ zero must be a member of a VSI. If it is needed to avoid mixed traffic and Flow Director Status indications on the same receive queue, then this VSI should not be the default VSI of the port and should not be a target of any switch filters.
- VFs and VMDq2 VSIs are allocated up to 16 scattered LQPs in the PF space by the *VPLAN_QTABLE* and *VSILAN_QTABLE* registers. The *VSILAN_QTABLE* registers enable scattered LQP allocation useful for dynamic VM motion.
 - For VFs, the *VSILAN_QTABLE* of all its VSIs must be set to the same indexes assigned to the VF by the *VPLAN_QTABLE*.
 - These tables are not readable for the VFs. Instead, the VF is notified by the PF about the number of its allocated LQPs (using a software API or the MailBox). The PF on its side, must allocate the LQPs starting at LQP zero and setting contiguous entries in these tables.
 - Mapping of a *VSILAN_QTABLE*[m] to a VF is the same for all other VSI registers defined by the *VSI_VSI2F* registers.
- VSIs allocated to normal PF traffic or PF control ports are expected to be statically allocated and therefore could be contiguous. Contiguous space is defined by the *VSIBASE* field in the *VSILAN_QBASE* register. VSIs that are allocated contiguous LQPs in the PF space can get any number of LQPs.
 - The size of the contiguous VSI is not enforced by hardware setting; software is responsible for keeping within the VSI boundaries.



- During reception, hardware verifies that the queue index generated by the receive classification filters does not exceed the PF queue range. However, it does not check to see if the queue index exceeds the VSI range. PF software is responsible for defining the receive classification filters so that the queues do not exceed the VSI space.
 - Contiguous vs. scattered VSI is controlled by the VSIQTABLE_ENA flag in the VSILAN_QBASE register.
 - PF software is responsible for defining free LQPs that are within the space of the PF.
 - TX and RX queues are allocated and configured in pairs.
 - If for any reason software configure and activate the RX queue only, it must also configure the QTX_CTL structure before enabling the RX queues.
- The QTX_CTL is part of the TX queue context, but it is used by the RX pipe as well.

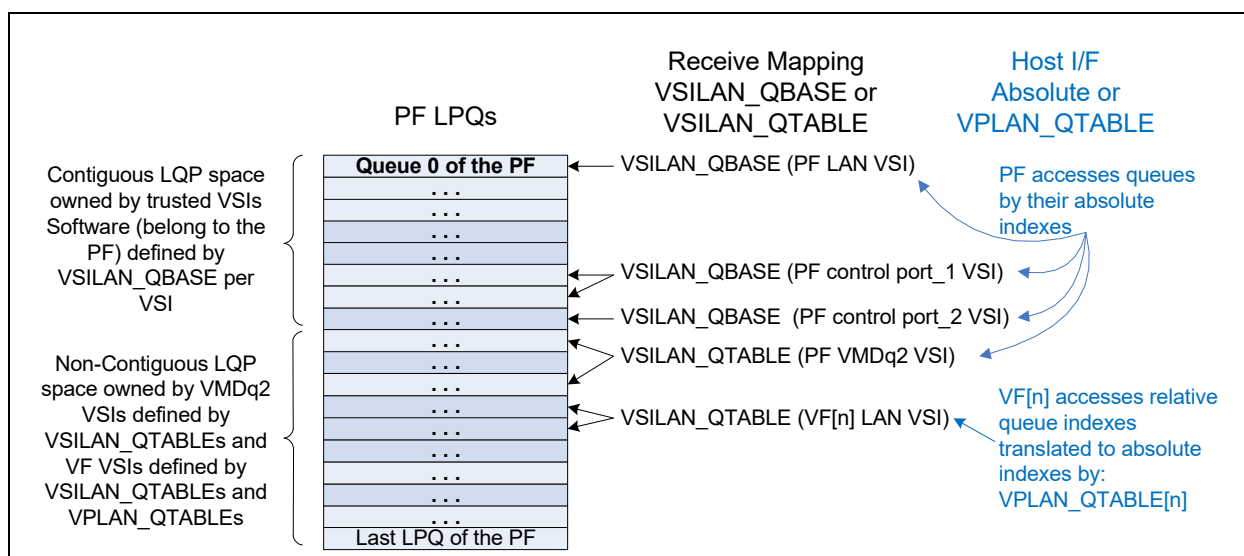


Figure 8-2. PF Queues Example

8.2.1.2.1 Software Access to the Queues of the Functions

PF software accesses its LQPs by their index within the PF space. Queue index 'm' of PF 'n' equals to 'm + PFLAN_QALLOC[n].FIRSTQ' in the device space. Accessing LQPs outside the PF space does not impact device functionality. Write accesses are terminated successfully on the PCIe bus with no impact on the hardware while read accesses provides meaningless data.

VF software accesses its LQPs using an index within the VF space. LQPs indexes are translated to indexes in the PF space by the VPLAN_QTABLE registers. The VPLAN_QTABLE supports up to 16 LQPs. A VF might get less than the maximum number of LQPs. A VF software attempt to access LQPs above the allocated ones does not impact device functionality (the same as described above for the PF).

Note: A function (PF or VF) might have multiple VSIs. Still, software accesses all LQPs by their index in the function space rather than by the index in the VSI spaces.



8.2.1.2.2 Associating Received Packets to VSI Queues

X710-TM4 associates received packets to VSIs by the embedded switch as described in [Section 7.4](#). It then associates packets to queues using the classification filters described in [Section 7.1](#).

- As opposed to the software interface, the classification filters assign a queue index in the VSI space. The queue index 'n' is mapped to the PF space as follows:
 - Queue index 'n' of a contiguous VSI is mapped to queue index 'n + VSILAN_QBASE.VSIBASE'
 - Queue index 'n' of a scattered VSI is mapped as follows:
 - For an even 'n' it is mapped to VSILAN_QTABLE[n/2].QINDEX_0
 - For an odd 'n' it is mapped to VSILAN_QTABLE[(n-1)/2].QINDEX_1
- Invalid receive queues
 - An invalid queue for contiguous VSI is identified if 'n + VSILAN_QBASE.VSIBASE' exceeds the PF queue space.
 - An invalid queue 'n' for scattered VSI is identified if its QINDEX in the VSILAN_QTABLE equals to 0x3FF or if 'n' is greater or equal than 16.
- Packets received to invalid queues are dropped and counted by the GLV_REPC counter of the VSI.

8.2.1.2.3 Dynamic Queue Allocation

X710-TM4 supports dynamic queue allocation between VSIs of each PF. Dynamic queue allocation is mainly aimed at supporting dynamic load balancing of VFs according to needs due to VM motion. Queues can be de-allocated from an active VSI during run time. These queues can then be allocated to another VSI of the same VF or to another VF. The whole flow is managed by the PF as detailed below.

The steps below relate to LQPs of VFs while it is similar to LQPs of VMs:

- PF software communicates to the VF that it needs to give up a specific LQP(s).
- The PF removes the queue(s) from the VPLAN_QTABLE and the VSILAN_QTABLE. As a result, the VF can no longer access the queue(s).
- PF software disables the relevant queues, following the queue disable flow described in [Section 8.3.3.1.2](#) and [Section 8.4.3.1.2](#). As part of the flow, the PF waits for the hardware indication that the queues are disabled with no further activity to host memory.
- The PF notifies the VF that it can release the host memory structures of the removed LQP(s).
- The PF remaps these LQP(s) to another VF as follows:
 - It programs the new Queue context parameters in the FPM and then enables these queues (see [Section 8.3.3.1.1](#) and [Section 8.4.3.1.2](#) for receive and transmit queue enablement flows).
 - It maps the LQP(s) to the new VF in the VPLAN_QTABLE and VSILAN_QTABLE and informs the VF about this action.
- The queues are ready to be used by the VF.

8.2.1.3 LAN Queue Pair Allocation Example

Allocate the LQPs to the VSIs of the PF by setting the VSILAN_QBASE registers, VSILAN_TABLE and VPLAN_TABLE. The table below lists an example of 5 VSIs while the PF owns 128 LQPs (as configured by PFLAN_QALLOC.FIRSTQ = 128 and PFLAN_QALLOC.LASTQ = 255):



VSI No'	VSI Usage	Requirement	Setting
none	N/A	1 LQP	Allocating dedicated receive queue for the Flow Director Filter Programming Status Descriptors. Queue index 0 in the PF space is absolute queue index 128
0	PF control port	1 LQP	VSILAN_QBASE[0].VSIQTABLE_ENA = 0 // contiguous range VSILAN_QBASE[0].VSIBASE = 1 Queue index 1 in the PF space is absolute queue index 129
1	PF LAN and its VMDq1 VMs	64 LQPs	VSILAN_QBASE[1].VSIQTABLE_ENA = 0 // contiguous range VSILAN_QBASE[1].VSIBASE = 2 Queue indexes 2...65 in the PF space are absolute queue index 130...193
2	PF traffic	16 LQPs	VSILAN_QBASE[2].VSIQTABLE_ENA = 0 // contiguous range VSILAN_QBASE[2].VSIBASE = 66 Queue indexes 66...81 in the PF space are absolute queue index 194...209
10	VMDq2	2 LQPs	VSILAN_QBASE[10].VSIQTABLE_ENA = 1 // scattered range VSILAN_QTABLE[10,0].QINDEX_0 and QINDEX_1 = 100, 102 and QINDEX_0 and QINDEX_1 in VSILAN_QTABLE[10,1...7] = 0x7FF Queues 100,102 in the PF space are absolute queues 228,230
100	VF[20]	4 LQPs	VSILAN_QBASE[100].VSIQTABLE_ENA = 1 // scattered range VSILAN_QTABLE[100,0].QINDEX_0 and QINDEX_1 = 90, 98 VSILAN_QTABLE[100,1].QINDEX_0 and QINDEX_1 = 110, 112 QINDEX_0 and QINDEX_1 in VSILAN_QTABLE[100,2...7] = 0x7FF VPLAN_QTABLE[20; 0,1,2,3,4:15].QINDEX = 90, 98, 110, 112, 0x7FF Queues 90,98,110,112 in the PF space are absolute queues 218,226,238,240

8.2.2 LAN Initialization Flow

This section describes the LAN engine initialization flow executed by the PF software driver. It is assumed that a PF reset (PFR) was initiated by the software prior to this flow:

- The LAN queue pairs (LQPs) are allocated to the PF by the NVM setting loaded to the PFLAN_QALLOC registers following a core reset (CORER).
- The statistic counters are cleared only at power on reset (POR). As part of the PF driver initialization, the software should read all the PF and its VF statistic counters. The values of these counters is the baseline for any statistics collected later.
- OS driver only step: Transit the device to non-PXE mode by initiating the Clear PXE Mode Admin Command. See the command description in [Section 8.2.2.1](#) and [Section 8.2.2.2](#).
- Most of the LAN engine logic is cleared by the hardware by core reset signal (CORER). If it is not guaranteed that a CORER was initiated, the software should clear the following control registers (listed in the LAN Transmit Receive Registers section) of the PF and its VFs:
 - LQPs mapping:
 - Clear the QTX_ENA, QRX_ENA, QTX_TAIL and QRX_TAIL registers of all the PF and its VFs LAN queue pairs.
 - Initialize LAN port parameters using the Set Port Parameters admin command (setting save bad packet, default VSI). In a single VSI per port, the VSI parameters are loaded from the NVM while software could execute this step only if it requires other settings, than the NVM



image. In MFP setup, these registers might be initialized only following the request of the first PF on this port.

- Allocate the LQPs to VFs and VSIs of the PF:
 - For each assigned VSI do the following as part of create VSI procedure:
 - Allocate the LQPs to the VSI (see programming example in [Section 8.2.1.3](#)).
 - Program the VSILAN_QBASE and VSILAN_QTABLE (use QBASE option for contiguous LQPs or the QTABLE option for scattered LQPs).
 - Program the VPLAN_QTABLE of each assigned VF.
- Enable individual receive and transmit queues of the PF and its VFs following the flow described in [Section 8.3.3.1.1](#) and [Section 8.4.3.1.1](#).

8.2.2.1 Clear PXE Mode Admin Command

The Clear PXE Mode Admin command transits the device from PXE to non-PXE mode. The structure of the Admin command and its completion are shown below. The complete software and device response is followed in section 8.2.2.2.

Table 8-1. Clear PXE Mode Admin Command (Opcode: 0x0110)

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.1 for details.
Opcode	2-3	0x0110	Command opcodes
Datalen	4-5	0x00	N/A (reserved zero)
Return value/VFID	6-7	0x00	N/A (reserved zero)
Cookie High	8-11	Cookie	Opaque value, will be copied by the firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value, will be copied by the firmware into the completion of this command
Reserved	16	0x2	Reserved 0x2
Reserved	17-31		

Table 8-2. Clear PXE Mode Admin Command Completion (Opcode: 0x0110)

Name	Bytes.Bits	Value	Remarks
Flags	0-1		See Section 7.10.5.2.2 for details.
Opcode	2-3	0x0110	Command opcode
Datalen	4-5	0x00	N/A
Return value/VFID	6-7		Some comments on specific errors (see Section 7.10.8 for the errors encoding): 0x0 - No Error 0xD - EEXIST (no action, the device is already in non-PXE mode)
Cookie High	8-11	Cookie	Opaque value, will be copied by the firmware into the completion of this command
Cookie Low	12-15	Cookie	Opaque value, will be copied by the firmware into the completion of this command
Reserved	16-31		Reserved



8.2.2.2 Transitioning Flow to Non-PXE Mode

8.2.2.2.1 Device Response to Clear PXE Mode Admin Command

1. When the Clear PXE Mode Admin command is initiated, the device checks the value of the PXE_MODE flag in the GLLAN_RCTL_0 register.
2. If the PXE_MODE flag is found cleared, then
 - a. Return a command completion with EEXIST indication.
3. Otherwise, the PXE_MODE flag is active
 - a. Disable Rx queue 0 and Rx queue 1 of all enabled PFs by clearing the QENA_REQ flag in the QRX_ENA[0] and QRX_ENA[1] registers of the PFs.
 - b. Wait until all receive queues are disabled by polling the QENA_STAT flag in the QRX_ENA registers of all above Rx queues. All Rx queues are disabled within a few microseconds.
 - c. Clear the PXE_MODE flag in the GLLAN_RCTL_0 register.
 - d. Flow is completed by posting a command completion with No Error.

8.2.2.2.2 Software Steps Transitioning to Non-PXE Mode

1. The admin queue must be active before the following steps.
2. Software initiates the Clear PXE Mode Admin command and waits for its completion.
3. Proceed with the software initialization flow.

8.2.3 Cloud Computing Support

The Cloud provides software, storage and compute infrastructure services over the Internet. Cloud platforms should provide isolation among tenants by creating private virtual networks over the physical network infrastructure and should be able to provision services on any physical machine within the data center. X710-TM4 provides filtering and pruning functionality that supports these goals.

Cloud providers build virtual network overlays over existing network infrastructure that provide tenant isolation and scaling. Tunneling layers added to the packets carry the virtual networking frames over existing Layer 2 and IP networks. Conceptually, this is similar to creating virtual private networks over the Internet. Processing these tunneling layers by the hardware is a critical element of this solution.

The tunneling packet formats supported by X710-TM4 are shown in [Figure 8-3](#). The fields in the packet used for switching functionality are described in [Section 7.4.9.3.1](#). The switch filters are described in subsections of [Section 7.4.4](#). See also transmit and receive descriptors for requested offloads on transmission and reported status on reception.

An overview of the packet processing is listed in [Table 8-3](#).



Table 8-3. Tunneled Packet Processing

Packet Type	Accept / Reject packet	Forward the packet to specific VSI (based on...)	Identify, Insert / Strip L2 Tags	L4/IPv4 XSUM	TSO	Tunneling Header Split (on top of non tunneling options)	RSS (hash) and FD Filters
IP in IP	Based on outer L2 MAC and optional L2 Tags (including VLAN, QinQ or E/S Tag w/ wo VLAN)	Outer IP	L2 Tags in the outer L2 Headers Only single VLAN in the inner L2 header	Both inner and outer IP headers, Inner L4 header	Yes	Outer IP header	Yes (same as non tunneled packets)
IP in GRE		Outer IP w/wo GRE key				GRE Header	
MAC in GRE		Inner L2 MAC/VLAN w/ wo one of the following: GRE key; Outer IP; Outer MAC				GRE Header	
MAC in UDP		Inner L2 MAC/VLAN w/ wo VN Key words				Outer UDP header and its extension header	

Figure 8-3 shows the supported tunneled packet formats:

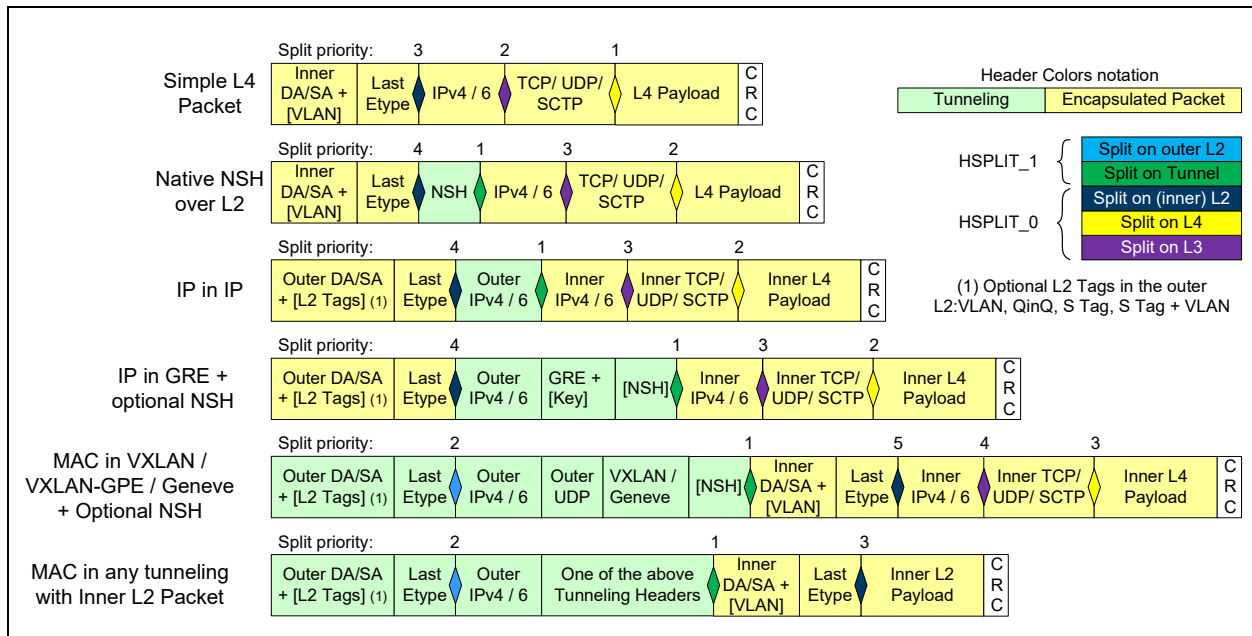


Figure 8-3. Tunneled Packet Formats

8.2.4 Steering Tag and Processing Hint (TPH) Support for LAN Engine Traffic

See Section 3.1.2.6.2 for information on Transaction Layer Packet (TLP) processing hint support. The following table describes how Steering tag and Processing hints are generated and how TPH operation is enabled for types of DMA traffic associated with the LAN queues.

**Table 8-4. Steering Tag and Processing Hint Programming by LAN Engine**

Traffic Access	Steering Tag Value and TPH Enablement	PH Value
Read receive Descriptor	CPUID and TPHRDesc in the Rx Queue Context	Desc_PH in GLTPH_CTRL register
Write back receive Descriptor	CPUID and TPHWDesc in the Rx Queue Context	Desc_PH in GLTPH_CTRL register
Write receive packet payload	CPUID and TPHData in the Rx Queue Context	DATA_PH in GLTPH_CTRL register
Write receive packet header	CPUID and TPHHead in the Rx Queue Context	DATA_PH in GLTPH_CTRL register
Read transmit Descriptor	CPUID and TPHRDesc in the Tx Queue Context	Desc_PH in GLTPH_CTRL register
Write back transmit Descriptor	CPUID and TPHWDesc in the Tx Queue Context	Desc_PH in GLTPH_CTRL register
Transmit Head write back	CPUID and TPHWDesc in the Tx Queue Context	Desc_PH in GLTPH_CTRL register
Read transmit packet	CPUID and TPHRPacket in the Tx Queue Context	DATA_PH in GLTPH_CTRL register

8.3 LAN Receive Data-Path

The LAN Receive Data-Path sections includes the following major topics:

- Receive packets stored in system memory.
- Indicating free descriptors to the hardware and indicating completed descriptors back to the software.
- Receive descriptor queues which are called also descriptor ring
- Receive arbitration
- Stateless Receive Offloads

8.3.1 Receive Packet in System Memory

Receive packets are posted to system (host) memory buffers indicated to the hardware by descriptors. There are several types of descriptors detailed in [Section 8.3.2](#); these include pointers to the data buffers and status indications of the received packets. The [Figure 8-4](#) shows two examples of receive packets in host memory composed of two buffers (indicated by two matched descriptors). X710-TM4 fetches the receive descriptors (on demand) to an internal cache. Here are a few rules relating to receive packet posting to host memory:

- Receive packets may span one to five buffers (descriptors).
- Receive packets shorter than 64 bytes are never posted to host memory (even in save bad frame mode - enabled by the SBP flag in the PRT_SBPVSI register). These packets are counted by the GLPRT_MSPDC counter per LAN Port.

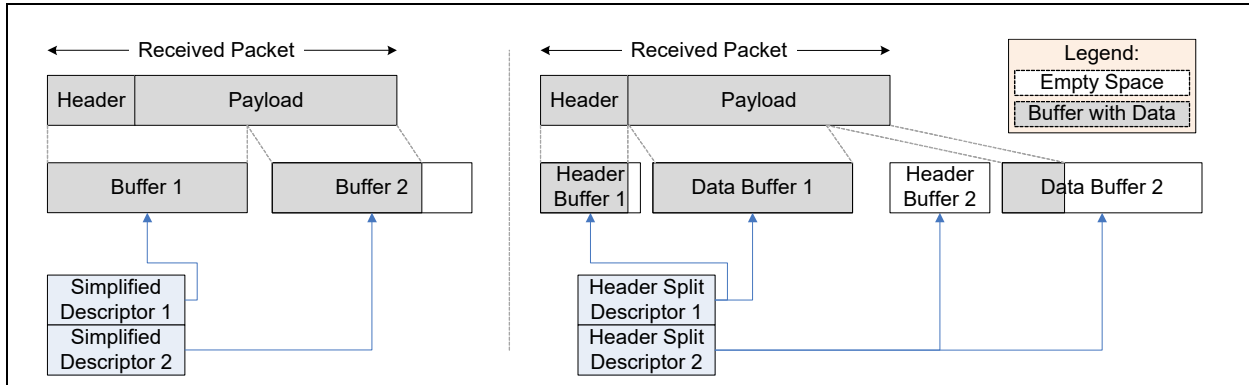


Figure 8-4. Receive Packet in System Memory

8.3.1.1 Receive Descriptor Cache

8.3.1.1.1 Descriptor Fetch Policy

To minimize PCIe and memory bandwidth overhead, X710-TM4 fetches multiple receive descriptors at once.; 8 or 4 descriptors when using 16 or 32 Byte descriptors respectively. New descriptors are fetched to the cache , when there are fewer descriptors than incoming packets require, or when the last free descriptor is used for a received packet.

Note the following rules relating descriptor fetch policy:

- Following a CORE Reset, the hardware wakes up in PXE mode (the PXE_MODE flag in the GLLAN_RCTL_0 is set). PXE mode functionality and limitations are as follows:
 - The receive queue should not be larger than 16 x 16 byte descriptors.
 - Software can bump the tail at descriptor granularity. Hardware fetches and writes back these descriptors at descriptor granularity as well.
 - Each packet may span only on a single buffer (in a single descriptor). A receive packet that is larger than a single buffer is reported as OVERSIZE in the receive descriptor.
- During nominal performance operation mode, the PXE_MODE flag must be cleared by the software. This step is expected to occur during the PF software initialization procedure. In case of multiple active PFs, only the first PF affects the device setting while the others do nothing.
- When the PXE_MODE flag is cleared, software should bump the tail at whole 8 x descriptors granularity. In this mode, hardware fetches descriptors in whole cache lines (4 x 32 byte descriptors or 8 x 16 byte descriptors).



Table 8-6. 32 Byte Receive Descriptors Read Format

Quad Word	6																																																																																																																															0
	3																																																																																																																															0
0	Packet Buffer Address																																																																																																																															
1	Header Buffer Address																																																																																																																															
2	Reserved (0x0)																																																																																																																															
3	Reserved (0x0)																																																																																																																															
	6																																																																																																																															0
	3																																																																																																																															0

The fields in first 16 Bytes Identical to the 16 Byte descriptors described in [Section 8.3.2.1.1](#).

8.3.2.2 Receive Descriptor - Write Back Format

The following subsections describe the fields of Receive Descriptor write back when using 16 byte and 32 byte descriptors. In both cases, a single packet might span on a single buffer or multiple buffers reported by their matched descriptors. If a packet is described by a single descriptor, then all the fields are valid. See below some rules that apply for a packet that is described by multiple descriptors:

- The following fields are valid in all descriptors of a packet: DD flag (Done); EOP flag (End of Packet) and PKTL field (Packet content length).
- The following fields are valid only in the first descriptor of a packet: HDRL (Packet content length in the header buffer); SPH (Header is identified for header split functionality) and HBO (Header Buffer overflow).
- All other fields are valid only in the last descriptor of a packet.

8.3.2.2.1 16 Byte Receive Descriptors WB Format

This section describes the 16 Bytes receive descriptor write back (WB) format.

Quad Word	6																									3	3																									1	1	1	1																									0																												
	3																									2	1																									6	5	4	3																									0																												
0	Filter Status												L2TAG1												rsv		MIRR																																																																																	
1	Length												PTYPE			rsv		Error			Status																																																																																							
	6																									3	3																									3	2	2	2																									1	1	1	1																									0
	3																									8	7																									0	9	7	6																									9	8	6	5																									0

RSV (Quad Word 1, 3 bits, Quad Word 0, 2 bits)

Reserved.

Status Field (Quad Word 1, 19 bits)



Bits	Name	Functionality
0	DD	Descriptor done indication flag.
1	EOP	End of packet flag is set to 1b indicating that this descriptor is the last one of a packet.
2	L2TAG1P	L2 TAG 1 presence indication while the L2 Tag is stripped from the packet and reported in the L2TAG1 field in the descriptor. The type of Tag is defined by the L2TSEL flag in the queue context. The L2TSEL flag selects between the first or second active flags in the SHOWTAG field in the VSI_TSR register of the VSI. The structure of this tag is defined by the matched GL_SWT_L2TAGxxx registers. If the specified L2 tag is not present in the packet the L2TAG1P flag is cleared.
3	L3L4P	For IP packets, this flag indicates that the L3 and L4 integrity check is processed by the hardware. In case of tunneled packet, the L3L4P relates to the inner IP header. Processing is according to the packet type which is reported in the PTYPE field. See Section 8.3.4.3 for a description of the supported integrity checks.
4	CRCP	CRCP indicates that the Ethernet CRC is posted with data to the host buffer. Note that strip CRC is enabled by the CRC Strip flag in the queue context. If the RXE error flag is set, the CRC bytes are not stripped regardless of the CRCStrip flag in the queue context. Loopback packets originated by another local VSI for which the hardware computes the CRC are never posted with the CRC bytes regardless of the CRCStrip setting in the queue context.
5:7	TSYNINDEX	Sampled 1588 timestamp index. <ul style="list-style-type: none"> bit 7 is the TSYNINDEX valid indication. Only when it is set to 1b are the lower 2 bits meaningful. bit 5:6 is the index of the PRRTSYN_RXTIME Register that holds the packet reception timestamp. Note that 1588 packets over UDP can be sampled regardless of a possible checksum error.
8	Reserved	Reserved
9:10	UMBCAST	Destination Address can be one of the following: <ul style="list-style-type: none"> 00b - Unicast 01b - Multicast 10b - Broadcast 11b - Mirrored Packet Non-parsed packets are indicated by PTYPE equals to PAYLOAD (non identified MAC header).
11	FLM	Flow director filter match indication. This flag is set if the received packet matches any of the Flow Director (FD) filters that direct the packet to a specific receive queue. See also the description of the RSSSTAT field below.
12:13	FLTSTAT	The FLTSTAT indicates the reported content in the Filter Status field. (See conditions in the Filter Status field): 00b - No Data in the filter status field (The packet does not meet any of the cases below) 01b - FD filter ID (this option is valid only for 16B descriptor while in 32B it is reported elsewhere) 10b - Reserved. 11b - Hash filter signature (RSS)
14	LPBK	Loopback indication, meaning that the packet originated from this system rather than the network.
15	IPV6EXADD	Set when an IPv6 packet contains a Destination Options Header or a Routing Header (See additional details in Section 8.3.4.3 .) If the packet contains two IPv6 headers (tunneling), the IPV6EXADD is a logic 'OR' function of the two IP headers.
16:17	Reserved	Reserved
18	INT_UDP_0	This flag is set for received UDP packets on which the UDP checksum word equals zero. Note: UDP checksum zero is an indication that there is no checksum. This option is valid only for IPv4 packets and considered an exception error for IPv6 packets (reported to the stack by the driver). Note: For tunneled packets with UDP header, this flag relates to the checksum field in the inner UDP header.



Error Field (Quad Word 1, 8 bits)

Table 8-7. Error Bits

Bits	Name	Functionality
0	RXE	The RXE error bit is an indication of any of the following MAC errors: CRC; Alignment; Oversize; Undersize; Length error. Packets with RXE are posted to host memory to receive queue 0 of the VSI defined by the PRT_SBPVSI register if enabled by the SBP flag in the same register. If the RXE flag is set then any other status fields reporting the content of the packet are meaningless.
1	RSV	Reserved
2	HBO	Header Buffer overflow. This flag is set when using Header split buffers or Split Always buffers and the identified packet header is larger than the header buffer. See Table 8-16 for details.
3:5	L3/L4E	For IP packets processed by the hardware the L3L4E flag has the following encoding: bit 3 - IPE: IP checksum error indication (for tunneled packets it is the most inner IP header indication). bit 4 - L4E: L4 integrity error indication (most inner L4 header in case of UDP tunneling). bit 5 - EIPE: External (most outer) IP header checksum error.
6	OVERSIZE	Oversize packet error indicates that the packet is larger than five descriptors in normal operation, or larger than one descriptor in PXE mode. In this case the portions of the packet that exceed the permitted number of descriptor(s) is not posted to host memory.
7	RSV	Reserved

MIRR (Quad Word 0, 14 Bits)

- Multiplexed field between Mirroring parameters as described in the following table.

Table 8-8. Mirroring Functionality

Bits	Name	Mirroring Functionality (UMBCAST field equals to 11b)
0:7	RSV	Reserved (equals to 0x7 for mirrored packets)
8:13	MIRR	Matched Mirror Rule ID that directed the packet to this queue.



L2TAG1 (Quad Word 0, 16 bits)

Stripped L2 Tag from the receive packet. This field is valid if the L2TAG1P flag in this descriptor is set (see additional description of the L2TAG1P flag).

Filter Status (Quad Word 0, 32 bits)

Multiplexed field between RSS (hash filter), FD Filter ID as indicated by the FLTSTAT field. Note that the FD filter ID is reported in this field only in 16 byte descriptors. In 32 byte descriptors, the data is reported in a different field.

- Otherwise, if the packet matches a FD filter that enables its FD filter ID reporting while using a 16 byte descriptor, then FLTSTAT equals 01b and this field contains the programmed FD filter ID.
- Otherwise, if the packet matches the Hash filter, then FLTSTAT equals 11b and this field contains the hash signature (RSS).
- Else, FLTSTAT equals 00b and this field is set to zero.

Length (Quad Word 1, 26 bits)

Bits	Name	Functionality
0:13	PKTL	Packet content length in the packet buffer defined in byte units.
14:24	HDRL	Packet content length in the header buffer defined in byte units.
25	SPH	The Split Header flag is an indication that the device identified the packet header. See Section 8.3.4.2 for a complete description of packet types identified for header split and conditions for usage of the header and data buffers.

PTYPE (Quad Word 1, 8 bits)

The Packet Type field encodes supported packet types as listed in [Table 8-9](#).

Note: In [Table 8-9](#), any UDP tunneling (Teredo, VXLAN) and GRE tunneling are reported as GRENAT.

Table 8-9. Packet Types

PTYPE	Description	PTYPE	Description
L2 Packet types			
0	Reserved	11	MAC, ARP
1	MAC, PAY2	12	
2	MAC, TimeSync, PAY2	13	
3	MAC, FIP, PAY2	14	
4	Reserved	15	
5	Reserved	16	
6	MAC, LLDP, PAY2	17	
7	Reserved	18	
8	Reserved	19	
9	Reserved	20	
10	MAC, EAPOL, PAY2	21	
Non Tunneled IPv4		Non Tunneled IPv6	
22	MAC, IPV4FRAG, PAY3	88	MAC, IPV6+IPV6FRAG, PAY3



Table 8-9. Packet Types (Continued)

PTYPE	Description	PTYPE	Description
23	MAC, IPV4, PAY3	89	MAC, IPV6, PAY3
24	MAC, IPV4, UDP, PAY4	90	MAC, IPV6, UDP, PAY4
25	Reserved	91	Reserved
26	MAC, IPV4, TCP, PAY4	92	MAC, IPV6, TCP, PAY4
27	MAC, IPV4, SCTP, PAY4	93	MAC, IPV6, SCTP, PAY4
28	MAC, IPV4, ICMP, PAY4	94	MAC, IPV6, ICMP, PAY4
IPv4 --> IPv4		IPv4 --> IPv6	
29	MAC, IPV4, IPV4FRAG, PAY3	36	MAC, IPV4, IPv6+IPV6FRAG, PAY3
30	MAC, IPV4, IPV4, PAY3	37	MAC, IPV4, IPV6, PAY3
31	MAC, IPV4, IPV4, UDP, PAY4	38	MAC, IPV4, IPV6, UDP, PAY4
32	Reserved	39	Reserved
33	MAC, IPV4, IPV4, TCP, PAY4	40	MAC, IPV4, IPV6, TCP, PAY4
34	MAC, IPV4, IPV4, SCTP, PAY4	41	MAC, IPV4, IPV6, SCTP, PAY4
35	MAC, IPV4, IPV4, ICMP, PAY4	42	MAC, IPV4, IPV6, ICMP, PAY4
IPv4 --> GRE/Teredo/VXLAN			
43	MAC, IPV4, GRENAT, PAY3		
IPv4 --> GRE/Teredo/VXLAN --> IPv4		IPv4 --> GRE/Teredo/VXLAN --> IPv6	
44	MAC, IPV4, GRENAT, IPV4FRAG, PAY3	51	MAC, IPV4, GRENAT, IPv6+IPV6FRAG, PAY3
45	MAC, IPV4, GRENAT, IPV4, PAY3	52	MAC, IPV4, GRENAT, IPV6, PAY3
46	MAC, IPV4, GRENAT, IPV4, UDP, PAY4,	53	MAC, IPV4, GRENAT, IPV6, UDP, PAY4,
47	Reserved	54	Reserved
48	MAC, IPV4, GRENAT, IPV4, TCP, PAY4	55	MAC, IPV4, GRENAT, IPV6, TCP, PAY4
49	MAC, IPV4, GRENAT, IPV4, SCTP, PAY4	56	MAC, IPV4, GRENAT, IPV6, SCTP, PAY4
50	MAC, IPV4, GRENAT, IPV4, ICMP, PAY4	57	MAC, IPV4, GRENAT, IPV6, ICMP, PAY4
IPv4 --> GRE/Teredo/VXLAN --> MAC			
58	MAC, IPV4, GRENAT, MAC, PAY3		
IPv4 --> GRE/Teredo/VXLAN --> MAC --> IPv4		IPv4 --> GRE/Teredo/VXLAN --> MAC --> IPv6	
59	MAC, IPV4, GRENAT, MAC, IPV4FRAG, PAY3	66	MAC, IPV4, GRENAT, MAC, IPv6+IPV6FRAG, PAY3
60	MAC, IPV4, GRENAT, MAC, IPV4, PAY3,	67	MAC, IPV4, GRENAT, MAC, IPV6, PAY3,
61	MAC, IPV4, GRENAT, MAC, IPV4, UDP, PAY4	68	MAC, IPV4, GRENAT, MAC, IPV6, UDP, PAY4
62	Reserved	69	Reserved
63	MAC, IPV4, GRENAT, MAC, IPV4, TCP, PAY4	70	MAC, IPV4, GRENAT, MAC, IPV6, TCP, PAY4
64	MAC, IPV4, GRENAT, MAC, IPV4, SCTP, PAY4	71	MAC, IPV4, GRENAT, MAC, IPV6, SCTP, PAY4
65	MAC, IPV4, GRENAT, MAC, IPV4, ICMP, PAY4	72	MAC, IPV4, GRENAT, MAC, IPV6, ICMP, PAY4
IPv4 --> GRE/Teredo/VXLAN --> MAC/VLAN			
73	MAC, IPV4, GRENAT, MACVLAN, PAY3		
IPv4 --> GRE/Teredo/VXLAN --> MAC/VLAN --> IPv4		IPv4 --> GRE/Teredo/VXLAN --> MAC/VLAN --> IPv6	
74	MAC, IPV4, GRENAT, MACVLAN, IPV4FRAG, PAY3,	81	MAC, IPV4, GRENAT, MACVLAN, IPv6+IPV6FRAG, PAY3
75	MAC, IPV4, GRENAT, MACVLAN, IPV4, PAY3,	82	MAC, IPV4, GRENAT, MACVLAN, IPV6, PAY3,
76	MAC, IPV4, GRENAT, MACVLAN, IPV4, UDP, PAY4	83	MAC, IPV4, GRENAT, MACVLAN, IPV6, UDP, PAY4



Table 8-9. Packet Types (Continued)

PTYPE	Description	PTYPE	Description
77	Reserved	84	Reserved
78	MAC, IPV4, GRENAT, MACVLAN, IPV4, TCP, PAY4	85	MAC, IPV4, GRENAT, MACVLAN, IPV6, TCP, PAY4
79	MAC, IPV4, GRENAT, MACVLAN, IPV4, SCTP, PAY4	86	MAC, IPV4, GRENAT, MACVLAN, IPV6, SCTP, PAY4
80	MAC, IPV4, GRENAT, MACVLAN, IPV4, ICMP, PAY4	87	MAC, IPV4, GRENAT, MACVLAN, IPV6, ICMP, PAY4
IPv6 --> IPv4		IPv6 --> IPv6	
95	MAC, IPV6, IPV4FRAG, PAY3	102	MAC, IPV6, IPv6+IPV6FRAG, PAY3
96	MAC, IPV6, IPV4, PAY3	103	MAC, IPV6, IPV6, PAY3
97	MAC, IPV6, IPV4, UDP, PAY4	104	MAC, IPV6, IPV6, UDP, PAY4
98	Reserved	105	Reserved
99	MAC, IPV6, IPV4, TCP, PAY4	106	MAC, IPV6, IPV6, TCP, PAY4
100	MAC, IPV6, IPV4, SCTP, PAY4	107	MAC, IPV6, IPV6, SCTP, PAY4
101	MAC, IPV6, IPV4, ICMP, PAY4	108	MAC, IPV6, IPV6, ICMP, PAY4
IPv6 --> GRE/Teredo/VXLAN			
109	MAC, IPV6, GRENAT, PAY3		
IPv6 --> GRE/Teredo/VXLAN --> IPv4		IPv6 --> GRE/Teredo/VXLAN --> IPv6	
110	MAC, IPV6, GRENAT, IPV4FRAG, PAY3	117	MAC, IPV6, GRENAT, IPv6+IPV6FRAG, PAY3
111	MAC, IPV6, GRENAT, IPV4, PAY3	118	MAC, IPV6, GRENAT, IPV6, PAY3
112	MAC, IPV6, GRENAT, IPV4, UDP, PAY4	119	MAC, IPV6, GRENAT, IPV6, UDP, PAY4
113	Reserved	120	Reserved
114	MAC, IPV6, GRENAT, IPV4, TCP, PAY4	121	MAC, IPV6, GRENAT, IPV6, TCP, PAY4
115	MAC, IPV6, GRENAT, IPV4, SCTP, PAY4	122	MAC, IPV6, GRENAT, IPV6, SCTP, PAY4
116	MAC, IPV6, GRENAT, IPV4, ICMP, PAY4	123	MAC, IPV6, GRENAT, IPV6, ICMP, PAY4
IPv6 --> GRE/Teredo/VXLAN --> MAC			
124	MAC, IPV6, GRENAT, MAC, PAY3		
IPv6 --> GRE/Teredo/VXLAN --> MAC --> IPv4		IPv6 --> GRE/Teredo/VXLAN --> MAC --> IPv6	
125	MAC, IPV6, GRENAT, MAC, IPV4FRAG, PAY3	132	MAC, IPV6, GRENAT, MAC, IPv6+IPV6FRAG, PAY3
126	MAC, IPV6, GRENAT, MAC, IPV4, PAY3	133	MAC, IPV6, GRENAT, MAC, IPV6, PAY3
127	MAC, IPV6, GRENAT, MAC, IPV4, UDP, PAY4	134	MAC, IPV6, GRENAT, MAC, IPV6, UDP, PAY4
128	Reserved	135	Reserved
129	MAC, IPV6, GRENAT, MAC, IPV4, TCP, PAY4	136	MAC, IPV6, GRENAT, MAC, IPV6, TCP, PAY4
130	MAC, IPV6, GRENAT, MAC, IPV4, SCTP, PAY4	137	MAC, IPV6, GRENAT, MAC, IPV6, SCTP, PAY4
131	MAC, IPV6, GRENAT, MAC, IPV4, ICMP, PAY4	138	MAC, IPV6, GRENAT, MAC, IPV6, ICMP, PAY4
IPv6 --> GRE/Teredo/VXLAN --> MAC/VLAV			
139	MAC, IPV6, GRENAT, MACVLAN, PAY3		
IPv6 --> GRE/Teredo/VXLAN --> MAC/VLAN --> IPv4		IPv6 --> GRE/Teredo/VXLAN --> MAC/VLAN --> IPv6	
140	MAC, IPV6, GRENAT, MACVLAN, IPV4FRAG, PAY3,	147	MAC, IPV6, GRENAT, MACVLAN, IPv6+IPV6FRAG, PAY3
141	MAC, IPV6, GRENAT, MACVLAN, IPV4, PAY3,	148	MAC, IPV6, GRENAT, MACVLAN, IPV6, PAY3,
142	MAC, IPV6, GRENAT, MACVLAN, IPV4, UDP, PAY4	149	MAC, IPV6, GRENAT, MACVLAN, IPV6, UDP, PAY4
143	Reserved	150	Reserved



Table 8-9. Packet Types (Continued)

PTYPE	Description	PTYPE	Description
144	MAC, IPV6, GRENAT, MACVLAN, IPV4, TCP, PAY4	151	MAC, IPV6, GRENAT, MACVLAN, IPV6, TCP, PAY4
145	MAC, IPV6, GRENAT, MACVLAN, IPV4, SCTP, PAY4	152	MAC, IPV6, GRENAT, MACVLAN, IPV6, SCTP, PAY4
146	MAC, IPV6, GRENAT, MACVLAN, IPV4, ICMP, PAY4	153	MAC, IPV6, GRENAT, MACVLAN, IPV6, ICMP, PAY4
Native NSH over L2 --> IPv4		Native NSH over L2 --> IPv6	
154	MAC, NSH, PAY2		
155	MAC, NSH, IPV4FRAG, PAY3	161	MAC, NSH, IPv6+IPV6FRAG, PAY3
156	MAC, NSH, IPV4, PAY3	162	MAC, NSH, IPv6, PAY3
157	MAC, NSH, IPV4, UDP, PAY4	163	MAC, NSH, IPV6, UDP, PAY4
158	MAC, NSH, IPV4, TCP, PAY4	164	MAC, NSH, IPV6, TCP, PAY4
159	MAC, NSH, IPV4, SCTP, PAY4	165	MAC, NSH, IPV6, SCTP, PAY4
160	MAC, NSH, IPV4, ICMP, PAY4	166	MAC, NSH, IPV6, ICMPv6, PAY4
Parser Aborted Code		Else	
255	Parser Aborted	Else	Reserved

8.3.2.2.2 32-byte Receive Descriptors WB Format

The 32-byte descriptor is composed of 4 quad words. The first two quad words are identical to the 16 byte descriptor write back other than the Flow Director filter ID that is moved from the Filter Status field to the FD Filter ID / Flexible Bytes High field.

Quad Word	6	4 4	3 3	3 3 3 2	2 2	1 1	1 1		
	3	8 7	8 7	2 1 0 9	7 6	6 5	4 3		0
0	Filter Status			L2TAG1		rsv	MIRR		
1	Length			PTYPE	rsv	Error	Status		
2	L2TAG2 (2nd)		L2TAG2 (1st)		rsv		rsv	Ext_Status	
3	FD Filter ID / Flexible Bytes High				Flexible Bytes Low				
	6	4 4	3 3	1 1		1 1			0
	3	8 7	2 1	8 7		2 1			



Ext_Status (Quad Word 2, 12 bits)

Bits	Name	Functionality
0	L2TAG2P	L2 TAG 2 presence indication while the L2 Tag is stripped from the packet to the L2TAG2 (1st) field in the descriptor. The type of the Tag is defined by the L2TSEL bit in the queue context. This flag selects between the first or second active flags in the SHOWTAG field in the VSI_TSR register of the VSI. The structure of this tag is defined by the matched GL_SWT_L2TAGxxx registers. If the stripped Tag is larger than one word, then the first word is posted to the L2TAG2 (1st) field and the second word is posted to the L2TAG2 (2nd) field. If it is a single word, it is stored in the L2TAG2 (2nd) field. If the specified L2 tag is not present in the packet the L2TAG2P flag is cleared.
1	RSV	Reserved
2:3	FLEXBL_STAT	The FLEXBL_STAT indicates the content of the Flexible Bytes Low field as follows: 00b - The field is set to zero 01b - Flexible first 4 bytes defined by the FD filter are posted in the Flexible Bytes Low 10b - Reserved 11b - Reserved
4:5	FLEXBH_STAT	The FLEXBH_STAT indicates the content of the FD Filter ID / Flexible Bytes High field as follows: 00b - The field is set to zero 01b - FD filter ID is reported in the FD Filter ID / Flexible Bytes High field 10b - Flexible second 4 bytes defined by the FD filter are posted in the FD Filter ID / Flexible Bytes High field 11b - Reserved
6:8	RSV	Reserved
9	FDLONGB	The FDLONGB flag is set if the matched FD filter's index within its bucket is above threshold. If the packet is searched in the FD filter and it is not found, the FDLONGB represents the bucket length relative to the same threshold. The threshold is defined by the MAXFDBLEN parameter in the GLQF_CTL register.
11	RSV	Reserved

L2 Tags (Quad Word 2, 32 bits)

Bits	Name	Functionality
0:15	L2TAG2 (1st)	Extracted L2 Tag 2 from the packet (see L2TAG2P flag).
16:31	L2TAG2 (2nd)	Extracted second word of the L2 Tag 2 from the packet (see L2TAG2P flag).

Flexible Bytes Low (Quad Word 3, 32 bits)

If the packet matches an FD filter that enables reporting flexible bytes from the packet, then FLEXBL_STAT equals 01b. Otherwise, FLEXBL_STAT equals 00b and this field is set to zero.

Bits	Name	FD filter flexible bytes (FLEXBL_STAT = 01b)
0:31	FLEXBL	This field contains 4 bytes from the Flexible Payload in the Field Vector (extracted from the packet). The word offset (in the Field Vector) of these bytes are defined by the programmed FLEXOFF parameter of the FD filter. The reported bytes in this field are defined in little endian notation while bits 0:7 are the LS byte which is the last byte on the wire.



Error Field (Quad Word 1, 6 bits)

Bits	Name	Functionality
0	Failed FD Programming	FD filter programming failed. This might be due to no space in the FD table, or because the function exhausted its quota.
1	Failed FD Removal	FD filter removal failed. This might be the result of an attempt to remove non-existent entry.
2:5	RSV	Reserved

Filter Status (Quad Word 0, 32 bits)

Equals the FD Filter ID for FD filter programming status.

8.3.3 LAN Receive Queue (Ring)

Received packets are posted to host memory through a set of queues. Each queue is a cyclic ring made of a sequence of receive descriptors in contiguous memory. These queues are also called descriptor rings. X710-TM4 supports up to 1536 receive queues allocated to PFs and VFs. See [Section 8.2](#). Receive queues are defined by a set of parameters called the queue context. The main parameters are the queue pointers shown in [Figure 8-5](#). The queue context includes additional parameters that define the queue functionality as detailed in [Section 8.3.3.2](#). The software interface to the queue for its initialization, during normal operation as well as queue disable flow, is described in [Section 8.3.3.1](#). X710-TM4 includes additional global setting option parameters for the whole device or per function that affect multiple queues described in [Section 8.3.3.1](#).

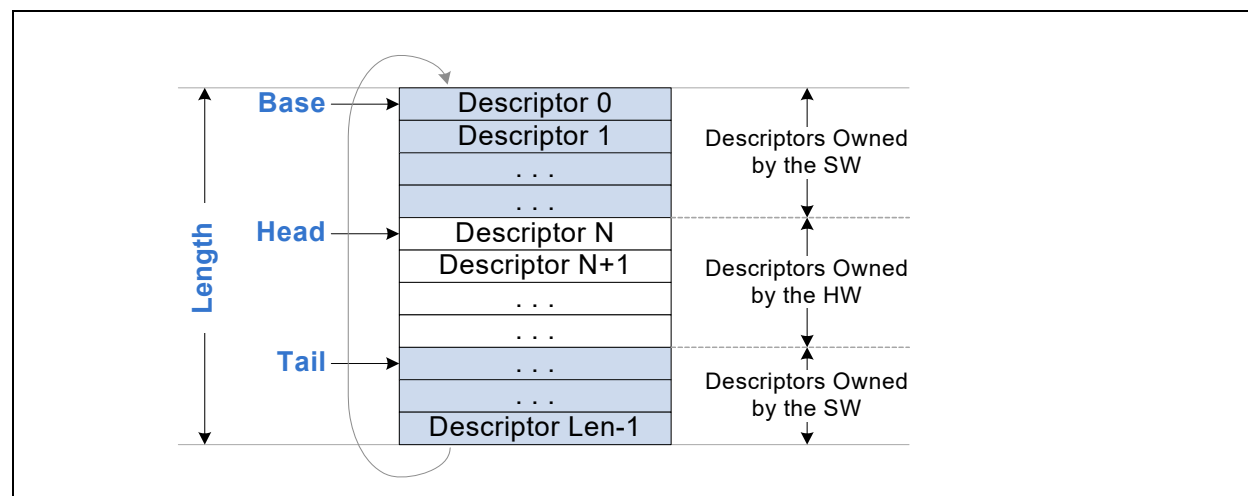


Figure 8-5. Receive Descriptor Ring Structure



8.3.3.1 Receive Queue Programming

Queue enable and disable flows are described in the following subsection and listed in the [Table 8-10](#) below.

Table 8-10. Receive Queue Enable / Disable Flags

QRX_ENA[n] register		Receive Queue State
QENA_REQ	QENA_STAT	
0	0	Queue is not enabled
1	0	Queue Enable request by the software.
1	1	Queue is enabled
0	1	Queue Disable request by the software.

8.3.3.1.1 Receive Queue Enable Flow

Queue enable flow is executed by the PF software for PF queues as well as for the queues of its VFs. The flow below assumes that the queue is already allocated as detailed in [Section 8.2.1.2](#).

- Software steps
- The function that owns the queue (PF or VF) allocates contiguous memory in its own memory space for the receive ring. It then programs the receive descriptors in the ring so they are ready for new packet reception.
- See examples for queue context setting in [Section 8.3.3.2.1](#).
- Clear the Tail pointer in the QRX_TAIL[n] register and then set the Tail pointer to the end of the descriptor ring ('n' is the queue index within the PF space).
- Set the QENA_REQ flag in the QRX_ENA[n] register ('n' is the queue index within the PF space).
- As a response, the hardware sets the QENA_STAT flag in the QRX_ENA[n] register. The QENA_STAT follows the QENA_REQ almost instantly and not more than 10 µsec after that. As soon as the QENA_STAT flag in the QRX_ENA[n] register is set, software can start using the queue.
- If the queue is targeted for a VF, PF software should also program the matched entry in the VFQ_TABLE. Then it is expected to inform the VF software that the queue is enabled. Note that the VFQ_TABLE is shared for the receive and transmit queues. So the PF software should enable the transmit queue before communicating to the VF.

8.3.3.1.2 Receive Queue Disable Flow

Queue disable flow is executed by PF software for PF queues as well as for the queues of its VFs.

- Remove the queue from the interrupt linked list as described in [Section 7.5.3.1.3](#).
- The PF software clears the QENA_REQ flag in the QRX_ENA[n] register, while 'n' is the queue index within the PF space.
- The hardware generates a queue disable marker to the receive pipe.
- Eventually, the queue disable marker gets to the top of the pipe. At this point, it is guaranteed that the pipe does not contain any additional receive packets for the specific queue.
- The hardware waits for completion of all outstanding requests from the specific queue on the PCIe bus.
- The queue context is invalidated and the QENA_STAT flag in the QRX_ENA[n] register is cleared.



- As soon as the QENA_STAT flag in the QRX_ENA[n] register is cleared, software can release all memory structures of the queue.

8.3.3.1.3 Fast Receive Queue Disable Flow

Fast queue disable flow should be executed by software only as part of a VF reset flow. Following a PFR, the device does this automatically.

- It is assumed that a VFR was initiated by the PF software and that the matched VFRD flag in the VPGEN_VFRSTAT register is already active. Or a VMR was initiated by PF software and the matched VMRD flag in the VSIGEN_RSTAT register is already active.
- The PF software sets the FAST_QDIS flag (and clears the QENA_REQ flag) in the QRX_ENA[n] register where 'n' is the queue index in the PF space for all VF or VM queues.
- The queue contexts are invalidated instantly from the internal cache, and the QENA_STAT flag in the matched QRX_ENA[n] register is cleared.

8.3.3.1.4 Software Fast Path Programming

8.3.3.1.4.1 Receive Descriptors and Tail bump

During normal operation, the function that owns the queue (PF or VF) accesses the hardware directly.

- Prepare receive descriptors by clearing the 'DD' bit and setting the buffer pointer(s). Start at the descriptor indicated by the TAIL pointer in the relevant QRX_TAIL register.
- The software should never set the TAIL to a value above the descriptors owned by the hardware minus 1. The descriptors considered owned by the hardware are those ones already indicated to the hardware but not yet reported as completed.
- Bump the TAIL to the last prepared descriptor plus one.
- The number of free descriptors owned by the hardware is defined by TAIL minus the HEAD. If the number of free descriptors becomes lower than LRXQTRESH, then an immediate interrupt is triggered. The HEAD and LRXQTRESH are listed in the [Table 8-11](#).

8.3.3.1.4.2 Receive Descriptor Reporting (Descriptor Write-Back)

X710-TM4 reports a completion of the receive packet in host memory using status indication in the receive descriptor(s) of the packet: the descriptor Write-Back (WB). X710-TM4 assigns the write-back completed descriptor status in one of the following cases:

- Whole line of descriptors (4x32 Byte descriptors or 8x16 Byte descriptors) completed.
- All completed descriptors of a queue whose context is invalidated from the internal cache.
- All completed descriptors of a queue whose interrupt is initiated by the internal core logic.

Most applications use interrupts to invoke the driver. Before initiating the interrupt, X710-TM4 posts all completed descriptors of the queue that might be kept in the device caches. Following an interrupt assertion, the device masks any further interrupts preventing interrupt nesting (as explained in [Section 7.5.1.4](#)). When interrupts are re-enabled (by the software), further interrupts that trigger additional write-back of completed descriptors can be initiated.

In some applications, X710-TM4 is activated in polling mode (with no interrupts). The following special settings should be used to enable the internal interrupt logic, so that the completed descriptors are posted to host memory:



- Enable interrupts internally in the QINT_RQCTL register of the queue, and map it to an MSI-X interrupt that is guaranteed to be inactive by the OS (all fields in the QINT_RQCTL should be programmed the same as done when operating with interrupts). One option could be an interrupt vector not exposed to the OS.
- The selected interrupt should be enabled by the INTENA flag in the matched xxINT_DYN_CTLN register.
- Interrupts are not initiated to the host, and the INTENA flag in the xxINT_DYN_CTLN register must not be auto-masked. Therefore the auto-masked option should be disabled globally for the whole device by setting the DIS_AUTOMASK_N flag in the GLINT_CTL register.
- The receive descriptors can be reported instantly for each packet by setting the ITR_INDX to NoITR. Another option is to enable some coalescing of reported descriptors using the ITR option. This option provides better PCIe and host memory bandwidth utilization at the expense of some possible latency of reporting the completed descriptor.
- Multiple receive queues can be associated with the same interrupt as described in [Section 7.5.3](#). In this mode, the transmit queues should not be associated with interrupts.
- The selected interrupt should point to the first receive queue in the linked list by setting the matched xxINT_LNKLSTN register (same programming as done when operating with interrupts).

8.3.3.2 Receive Queue Context Parameters

This section describes setting options for LAN receive queue parameters called the queue context.

- Queue enablement flags in the QRX_ENA[n] registers while 'n' is the queue index of the PF. Queue enable and disable flow by the PF are explained in [Section 8.3.3.1](#).
- The TAIL pointer in the QRX_TAIL[n] registers for the PF and VFQRX_TAIL[n] registers for the VFs while 'n' is the queue index of the function (PF or VF). The usage of the Tail register is detailed in [Section 8.3.3.1.3](#).
- Interrupt related registers QINT_RQCTL[n] while 'n' is the queue index of the PF. The usage of these registers is described in [Section 7.5.2.2](#).

Table 8-11. LAN Rx Queue Context in the Private Host Memory(174 bits = 25 Byte)

Alias	Width [bits]	LS bit	MS bit	Type	SW Init	Description
HEAD	13	0	12	Dynamic	0x0	Receive Queue Head. An index relative to the beginning of the queue that defines the next descriptor to be used. During idle time, all descriptors starting by the HEAD up to (excluding) the RTAIL are owned by the hardware and the rest are owned by software (as shown in Figure 8-1). During dynamic operation it is not guaranteed that all descriptors below the HEAD are completed.
CPUID	8	13	20	Dynamic	0x0	CPU Socket ID for TPH. The CPU socket ID is updated by the hardware.
RSV	11	21	31	N/A	0x0	Reserved
BASE	57	32	88	Static	BASE	Receive Queue Base Address.- Indicates the starting address of the descriptor queue defined in 128 Byte units.



Table 8-11. LAN Rx Queue Context in the Private Host Memory(174 bits = 25 Byte)

Alias	Width [bits]	LS bit	MS bit	Type	SW Init	Description
QLEN	13	89	101	Static	QLEN	Receive Queue Length.- Defines the size of the descriptor queue in descriptors units from 8 descriptors (QLEN=0x8) up to 8K descriptors minus 32 (QLEN=0x1FE0).- QLEN restrictions: When the PXE_MODE flag in the GLLAN_RCTL_0 register is cleared, the QLEN must be whole number of 32 descriptors. When the PXE_MODE flag is set, the QLEN can be one of the following options: Up to 4 PFs, QLEN can be set to: 8, 16, 24 or 32 descriptors Up to 8 PFs, QLEN can be set to: 8 or 16 descriptors Up to 16 PFs, QLEN can be set to: 8 descriptors
DBUFF	7	102	108	Static	DBUFF	Receive Packet Data Buffer Size. The Packet Data Buffer Size is defined in 128 byte units. Must be at least 1K bytes and up to 16K minus 128 bytes.
HBUFF	5	109	113	Static	HBUFF	Receive packet Header Buffer Size.- The Header Buffer Size is defined in 64 byte units enabling buffer size up to 2KB minus 64B.
DType	2	114	115	Static	DType	Descriptor Type as defined in Table 8-13.
DSize	1	116	116	Static	DSize	Descriptor Size flag.- '0' for 16B descriptors and '1' for 32B descriptors
CRCStrip	1	117	117	Static	CRCStrip	CRC Strip.- Strip the Ethernet CRC bytes before the packet is posted to host memory. Note that no CRC Strip option may work properly only if the whole packet is posted to the data buffer(s) in host memory with no other strip option.
L2TSEL	1	119	119	Static	0x0	The L2TSEL defines the reported L2 Tags in the receive descriptor. When the L2TSEL flag is cleared, the second L2 tag defined by the SHOWTAG field in the VSI_TSR register of the VSI is posted to the L2TAG1 field. The first tag is reported in the L2TAG2 field.- When the L2TSEL flag is set to 1b, the above L2 tags are switched between the L2TAG1 and L2TAG2 fields.
HSPLIT_0	4	120	123	Static	HSPLIT_0	Header Split 0 control as described in Table 8-14
HSPLIT_1	2	124	125	Static	HSPLIT_1	Header Split 1 control as described in Table 8-15
RSV	1	126		N/A	0x0	Reserved
RSV	46	132	173	Static	0x0	Reserved
RXMAX	14	174	187	Static	RXMAX	Max packet size for this queue defined in byte units. The <code>rxmax</code> parameter defines the whole packet size starting at the L2 header up to including the Ethernet CRC. The RXMAX must not be set to a larger value than 5 x DBUFF (since receive packet must never span on more than 5 buffers). Received packet larger than RXMAX is dropped and counted by the GLV_REPC counter of the VSI. Note that packets larger than MFS defined per MAC are dropped by the MAC even before it gets to the receive queue.
RSV	5	188	192	Static	0x0	Reserved
TPHRDesc	1	193	193	Static	TPHRDesc	Read Descriptor TPH Ena (descriptor fetch).
TPHWDesc	1	194	194	Static	TPHWDesc	Write Descriptor TPH Ena (descriptor writeback).
TPHData	1	195	195	Static	TPHData	Packet Data TPH Ena.
TPHHead	1	196	196	Static	TPHHead	Packet Header TPH Ena.
RSV	1	197	197	Static	0x0	Reserved
LRXQTRESH	3	198	200	Static	LRXQTRESH	Low Receive Queue Threshold defined in 64 descriptors units. When the number of free descriptors (defined by Tail minus Head) goes below the LRXQTRESH an immediate interrupt is triggered.
RSV	55	201	255	Static	0x0...01	Reserved



8.3.3.2.1 Examples for Receive Queue Context Setting

Listed below are several examples for receive queue context settings.

Control Port Receive Queue Context Example =

	7	6	5	4	3	2	1	0
Context bit index	255 ... 224	223 ... 192	191 ... 160	159 ... 128	127 ... 96	95 ... 64	63 ... 32	31 ... 0
Init Hex Value	00000000	0000021E	01800000	00000000	00200301	00000000	001579A0	00000000

BASE = 0x001579A0	HBUF = 0	HSPLIT = 0 (no split)	LRXQTRESH = 0 (no thresh.)
QLEN = 0x80 (128 desc.)	DType = 00b (no split)	RXMAX = 0x600 (1536)	CRCStrip = 1
DBUFF = 12 (1536 bytes)	DSize = 0 (16 bytes)	TPH = 0xF (enable all flags)	FCENA = 0

BASE = 0x00BACD000	HBUF = 0	HSPLIT = 0 (no split)	LRXQTRESH = 2 (128 desc.)
QLEN = 0x400 (1024 desc.)	DType = 00b (no split)	RXMAX = 0x864 (2148 bytes)	CRCStrip = 1
DBUFF = 17 (2176 bytes)	DSize = 1 (32 bytes)	TPH = 0xF (enable all flags)	FCENA = 1

8.3.3.2.2 Examples for Receive Buffer Size Setting

Table 8-12 lists the most common used buffer size settings used by LAD drivers.

Table 8-12. Common Buffer Size Settings Used by LAD Drivers

OS	Queue type	HSPLIT	DBUFF	HBUF	Notes
Linux	Standard	0x0	1536	-	
	Standard	enabled	2048	256	
	Jumbo	0x0	multiple of 1K	-	unlikely to use this configuration
	Jumbo	enabled	2048	256	
Windows	Any	0x0	multiple of 1K	-	up to 10K
	VMQ	enabled	2048	64	L2 header split
FreeBSD	Any	0x0	2K, 4K, 8K, or 10K	-	
	Any	enabled	2K, 4K, 8K, or 10K	256	
ESX NPA	Any	enabled	4096	1984	max header buffer
Solaris	Any	0x0	256, 1K, 2K	-	
XEN					same as Linux
KVM					same as Linux



8.3.4 Stateless Receive Offloads

8.3.4.1 Strip Ethernet CRC bytes

See [Section 7.2.1.2](#).

8.3.4.2 Header Split

This feature consists of splitting a received packet into two separate regions based on the packet content. Splitting is usually between the packet header that can be posted to a dedicated buffer and the packet payload that can be posted to a different buffer (or multiple buffers). The size of these buffers is defined by the DBUFF and HBUFF parameters in the receive queue context. This kind of splitting is useful when different buffer allocation rules may apply to these buffers or different rules for TPH enablement. Header split is enabled per receive queue by the DTYPE and HSPLIT_0 and HSPLIT_1 fields in the receive queue context as listed in [Table 8-13](#), [Table 8-14](#) and [Table 8-15](#), and illustrated in [Figure 8-6](#). The split between the header buffer, the payload buffers and the status reporting is listed in [Table 8-16](#). The physical pointers to the header and payload buffers are defined in the receive descriptor.

These rules concern the header split:

- A packet that has a MAC error (reported by the RXE flag) is posted as whole to the packet buffers with no split. Note that posting such packets to the host is enabled on in Save Bad Frame mode (enabled by the SBP flag in the PRT_SBPVSI register).
- For tunneled packets, the rules defined by HSPLIT_1 parameter take precedence over those defined by the HSPLIT_0 parameter. This means that if any flag in HSPLIT_1 is enabled and the packet matches that setting, then the packet is split according to HSPLIT_1 regardless of HSPLIT_0 setting.
- In each individual register: HSPLIT_1 or HSPLIT_0, the packets are split according to the lowest matched entry in the following tables. If both HSPLIT_0 and HSPLIT_1 are set to zero, then DTYPE in the receive queue context must be set to 00b. Otherwise (any header split is enabled by these registers), the DTYPE must be set to one of the header split options: 01b or 10b (explained in the following subsections).
- If the packet is posted to multiple descriptors, only the header buffer of the first one is used.
- The packet header cannot span buffers. If the header buffer is smaller than the received header, the header is posted together with the packet payload. See [Table 8-16](#).
- The header of a fragmented IP packet is defined up to and including the IP header, regardless of whether the fragment includes the L4 header. For IPv6 headers, the IP header is defined up to and including the fragmented extension header.
- When a packet is replicated to multiple receive queues, the packet can be split differently on each according to the queue settings.
- Header split is supported for packets received from the network as well as local VM to VM traffic.

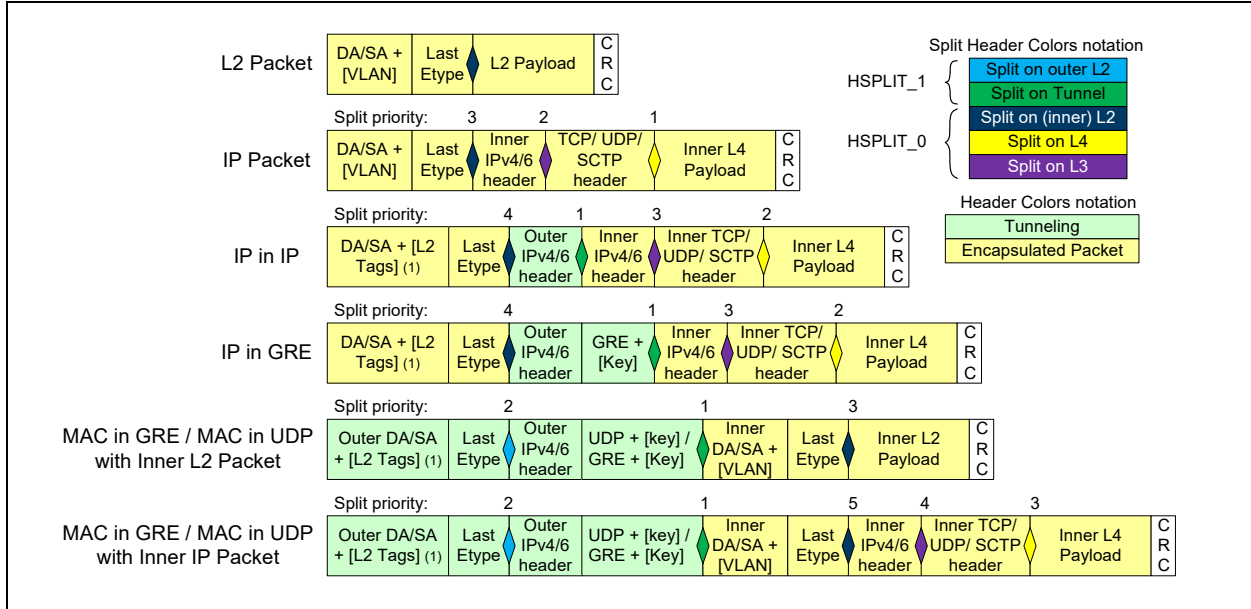


Figure 8-6. Header Split Option by Packet Formats

Table 8-13. Header Split Modes Defined by the DTYPE Field

DTYPE	Functionality
00b - Single buffer descriptors	No header split mode
01b Header split descriptors	Header split is enabled for packets that the hardware identifies their headers as enabled by the HSPLIT field. The packet content up to including the most inner header enabled by the HSPLIT is posted to the header buffer. The rest of the packet is posted to the payload buffer. See Table 8-16 for rules of the header and payload buffers usage.
10b Split always	Header split always is enabled regardless of the HSPLIT setting. If the packet header is identified as defined by the HSPLIT, the packet content up to including the most inner header enabled by the HSPLIT is posted to the header buffer. See Table 8-16 for rules of the header and payload buffers usage.
11b	Reserved

Table 8-14. Split Headers Enabled by the HSPLIT_0 Field

HSPLIT_0	Functionality
0000b	No Header are enabled by HSPLIT_0.
xxx1b	Enable split after L2 header. In case of L2 tunneling it is the second (inner) L2 header.
xx1xb	Enable split after the IP header. In case of tunneling it is the second IP header In case of option/extension IP headers the split is after these headers
x1xxb	Enable split after the UDP and TCP header (in case of UDP tunneling it is the second UDP header).
1xxxb	Enable split after the SCTP header.

**Table 8-15. Split Headers Enabled by the HSPLIT_1 Field**

HSPLIT_1	Functionality
00b	No split on tunneling headers.
x1b	Enable split after the outer (tunneling) L2 header.
1xb	Enable split on the tunneling header as follows: Non tunneled packet - No impact IP in IP - Enable split after the outer IP header IP in GRE or MAC in GRE - Enable split after the GRE header MAC in UDP - Enable split after the tunneling UDP header (including its private UDP header)

Table 8-16. Header Split vs. Packet and Buffer Sizes

DTYPE	Condition	Header and Payload DMA	SPH	HBO	PKTL (1)	HDRL (1) (2)
00b - Single buffer	None	Post the whole packet to the packet buffer(s)	0	0	Size of the whole packet	0x0
01b - Header split	Header is not identified	Post the whole packet to the packet buffer(s)	0	0	Size of the whole packet	0x0
	Header size <= HBUFF (2)	Post header to header buffer and payload to the packet buffer(s)	1	0	Size of the packet payload	Header size
	Header size > HBUFF (2)	Post the whole packet to the packet buffer(s)	1	1	Size of the whole packet	Header size
10b Split Always	Packet length <= HBUFF	Post the whole packet to the header buffer(s)	0	0	0x0	Packet length
	Header is not identified and packet length > HBUFF	Post the whole packet to the header buffer + packet buffer(s)	0	0	Size of the whole packet minus HBUFF	HBUFF
	Header size <= HBUFF	Post header to the header buffer and the payload to the packet buffer(s)	1	0	Size of the packet payload	Header size
	Header size > HBUFF	Post the whole packet to the header + packet buffer(s)	1	1	Size of the whole packet minus HBUFF	Header size

Note:

- If the data posted to the packet buffer is larger than PKTL, multiple buffers (descriptors) are used.
 - All buffers but the last one are full (PKTL = DBUFF) while the last one contains the rest of the data.
 - Only the header buffer of the first descriptor is used.
- For the sake of the split conditions, the packet and header lengths are taken from the received packets including all L2 tags (VLAN for example) as well as the CRC bytes. Regardless if these tags are posted to the buffers or posted to the receive descriptors or extracted from the packet. Orthogonal to this condition, the reported HDRL parameter represents the actual data posted to the buffer.

8.3.4.3 Receive L3 and L4 Integrity Check Offload

The X710-TM4 offloads the following L3 and L4 integrity checks: IPv4 header(s) checksum, Inner TCP or UDP checksum and SCTP CRC integrity (see [Table 8-17](#)). The X710-TM4 identifies the packet type and then checks the matched integrity scheme. The identified packet type is reported on the PTYPE field in the receive descriptor. Processing indication of the L3 and L4 headers is reported on the L3L4P flag in the receive descriptor. Potential IPv4 checksum errors, L4 integrity errors, and outer IPv4 checksum errors are reported by the IPE, L4E and EIPE error flags in the receive descriptor respectively.



Some rules for integrity check offload are listed here. If these rules are not met, integrity offload is not provided and the L3L4P is not set.

- IPv4 header is assumed to be at least 20 bytes long (the length of the basic header).
- IPv4 headers might have any IP option headers that fit within the maximum header size (60 bytes).
- IPv6 support: The pseudo header for the L4 checksum takes into account the addresses in the IPv6 header, ignoring the optional extension headers. Packets with Routing Header type 2 and Destination Options Header with Home Address option contain an alternative IP address in the extension header. Therefore, checksum calculation for such packets probably results in an erroneous value. X710-TM4 indicates the existence of a Destination Options Header or a Routing Header in the IPV6EXADD bit of the RX descriptor. Software can then do one of the following:
 - Ignore the checksum done by the device.
 - Parse the extension header and determine whether it contains an IP address. Then, only in this case, ignore the checksum done by the device.
- Fragmented packets - X710-TM4 parses fragmented receive packets up to and including the IP header (for IPv4) or up to and including the fragmentation extension header (for IPv6):
 - L4 checksum offload is not supported for IPv6 fragmented packets, and the L3L4P flag in the receive descriptor is not set.
 - A fragmented IPv4 packet is offloaded up to and including the IP header.
- TCP header is assumed to be at least 20 bytes long (the length of the basic header).
- The TCP header might have any optional headers that fit within the maximum header size (60 bytes).
- VM to VM loopback traffic is processed by the hardware for L3/L4 integrity check as any other packet received from the network.

Table 8-17 lists all supported packet formats and the processed integrity. The table uses the following notations:

- IP is a generic term for IPv4 header or IPv6 header. The IPv4 header can have IP option headers and the IPv6 header can have IPv6 extension headers.
- L4 is a generic term for UDP, TCP or SCTP headers.
- IP checksum is meaningful only for IPv4.
- Checksum is a generic term for UDP and TCP checksum as well as SCTP CRC integrity.
- Zero UDP checksum: Zero UDP checksum for IPv4 packet is treated as no checksum and is reported by the hardware as no error. Zero UDP checksum for IPv6 packet is illegal and is reported by the hardware as L4 checksum error.

Table 8-17. Integrity Offload Check for Receive Packet Types

Packet Type	Supported Integrity Offload	Reported L3L4P
IP -> [data / Unknown / fragmented]	IP checksum offload	1 (for IPv4) / 0 (for IPv6)
IP -> L4	IP and L4 checksum offload	1
IP -> IP -> [data / Unknown / fragmented]	2 x IP checksum offload	1 if at least one of the IP headers is IPv4
IP -> IP -> L4	IP and L4 checksum offload	1
IP -> [tunnel header] -> IP -> data / Unknown / fragmented	Only IP checksum offload.	1 if at least one of the IP headers is IPv4
IP -> [tunnel header] -> IP -> L4	IP and L4 checksum offload (1)	1
IP -> [tunnel header] -> data and IP -> [tunnel header] -> MAC -> data		1 (for IPv4) / 0 (for IPv6)

**Table 8-17. Integrity Offload Check for Receive Packet Types**

Packet Type	Supported Integrity Offload	Reported L3L4P
IP -> [tunnel header] -> MAC -> IP -> data	IP checksum (relevant only for IPv4)	1 if at least one of the IP headers is IPv4
IP -> [tunnel header] -> MAC -> IP -> L4	IP and L4 checksum offload (1)	1
(1) The L4 checksum offload relates to the inner header: <ul style="list-style-type: none"> - For UDP or TCP protocols, the hardware calculates the expected checksum including the pseudo IP header - For SCTP protocol, the hardware calculates the expected SCTP CRC (2) Tunneling headers could be one of the following: GRE, Teredo, VXLAN UDP header		

8.4 LAN Transmit Data-Path

The LAN Transmit Data-Path section covers the following major topics:

- Transmit packets stored in system memory and indicated to the hardware by descriptors
- Transmit descriptor queues which are called also descriptor rings and Transmit arbitration queue lists
- Stateless transmit offloads

8.4.1 Transmit Packet in System Memory

Transmit packets are made up of data buffers in host memory indicated to the hardware by descriptors (16 byte structures described in [Section 8.4.2.2](#)). These descriptors include pointer and length pairs to the data buffers as well as control fields for the transmit data processing. In some cases, additional control parameters that cannot fit within the data descriptors are needed to process the packet(s). In this case, additional context descriptor(s) are needed in front of the data descriptors. A few examples for needed context descriptor(s) are: Transmit Segmentation Offload (TSO), FD filter programming. [Figure 8-8](#) shows an example of a transmit packet in host memory composed of 2 buffers (header buffer and payload buffer), indicated by 2 matched data descriptors and optional context descriptor.

Here are a few rules related to the transmit packet in host memory:

- The total size of a single packet in host memory must be at least 17 bytes and up to the Max Frame Size of the port as configured by the Set MAC Config admin command.
 - Packets outside this range are considered malicious. The respective queue is stopped and an interrupt is issued to the PF as described in [Section 7.6.2.2.1](#). The relevant event is Bad Single Send size.
 - This rule applies for single packet send as well as any packet within a transmit segmentation (TSO).
- A single transmit packet may span up to 8 buffers (up to 8 data descriptors per packet including both the header and payload buffers).
- The total number of data descriptors for the entire TSO is unlimited as long as each segment within the TSO obeys the previous rule (up to 8 data descriptors per segment for both the TSO header and the segment payload buffers).
- If a packet or TSO spans multiple transmit data descriptors, the fields in all the data descriptors must be valid.



8.4.2 Transmit Descriptors

This section describes the descriptors provided on the LAN transmit queues. The X710-TM4 supports the following Transmit descriptor types.

LAN Descriptors: In many cases only the data descriptor is needed to indicate a packet. Multiple descriptors are needed for FD filter programming, for TSO and/or packets that requires offloads like tunneling. In those cases, the order of the descriptor must follow the order in the [Table 8-18](#) below.

Table 8-18. LAN Descriptor Types

Type	DTYP value	Description	Reference
Optional NOP Descriptor(s)	0x1	NOP descriptors can be used to align descriptors to cache lines (optional usage)	Section 8.4.2.1.2
LAN Transmit Context Descriptor	0x1	Used as a companion to the Transmit Data descriptor to provide more information on the packet	Section 8.4.2.2
FD Filter programming descriptor	0x8	Used to program flow director filters	Section 8.4.2.3
Transmit Data Descriptor	0x0	Regular data descriptor used to send LAN packets	Section 8.4.2.1

The following sections describe these descriptors.

Note: For all descriptors: fields indicated as 'rsv' (reserved) should be set to zero by the software at programming time.

8.4.2.1 General Descriptors

8.4.2.1.1 Transmit Data Descriptor

Quad Word	6																	0
	3																	0
0	Tx Packet Buffer Address																	
1	L2 Tag 1				Tx Buffer Size				Offset				CMD				DTYP	
	6	4	4	3	3	1	1											
	3	8	7	4	3	6	5					4	3	0				



Descriptor Type - DTYP (Quad Word 1, bits 0:3)

Command Field - CMD (Quad Word 1, bits 4:15)

Table 8-19. Transmit Command Field Descriptors

Bits	Name	Functionality
0	EOP	End Of Packet. The EOP flag is set in the last descriptor of a packet or TSO.
1	RS	Report Status. When set, the hardware reports the DMA completion of the transmit descriptor and its data buffer. Completion is reported by descriptor WB or by head write back as configured by the HEAD_WBEN flag in the transmit context. When it is reported by descriptor WB, the DTYP field is set to 0xF and the RS flag is set. The RS flag can be set only on a last Transmit Data Descriptor of a packet or last Transmit Data Descriptor of a TSO or last Transmit Data Descriptor of a filter.
2	RSV	Reserved, must be set to 1b.
3	IL2TAG1	Insert an L2 tag from the L2TAG1 field in this descriptor. For MAC in UDP or MAC in GRE tunneling, the L2TAG1 field is relevant to the outer L2 header. See IL2TAG_IL2H for VLAN tag insertion to the inner L2 header. The type of the Tag is defined by the L2TAG1INSERTID field in the VSI_L2TAGSTXVALID register of the VSI. The structure of this tag is defined by the matched GL_SWT_L2TAGxxx registers. If the type of the L2 Tag requires more than 2 variable bytes, then additional bytes are taken from the L2TAG2 field while the L2TAG1 is first on the wire. In this case, the IL2TAG2 flag must be cleared.
4	DUMMY	When the Dummy flag is set, the packet is not transmitted (internal and external). The Dummy indication can be useful for context programming purposes: FD filters. Note that when using the Dummy option, the packet does not have to have a correct checksum. Software should set all the fields in the data descriptor and context descriptors describing the packet structure as it does for nominal packets.
5:6	IIPT	The IP header type and its offload. In case of tunneling, the IIPT relates to the inner IP header. See also EIPT field for the outer (External) IP header offload. 00 - non IP packet or packet type is not defined by software 01 - IPv6 packet 10 - IPv4 packet with no IP checksum offload 11 - IPv4 packet with IP checksum offload
7	Reserved	Reserved
8:9	L4T	L4T is the L4 packet type: 00b - Unknown / Fragmented Packet 01b - TCP 10b - SCTP 11b - UDP When the L4T is set to other values than 00b, the L4LEN must be defined as well. When set to UDP or TCP, the hardware inserts the L4 checksum and when set to SCTP the hardware inserts the L4 CRC.
10:11	RSV	Reserved



Header Offset Parameters - OFFSET (Quad Word 1, bits 16:33)

Table 8-20. Header Offset Parameters

Bits	Name	Functionality
0:6	MACLEN	MAC Header Length defined in Words. In case of a tunnel packet, MACLEN defines the outer L2 header. MACLEN defines the L2 header length up to including the Ethertype. If L2 tag(s) are provided in the data buffers, then they are included in MACLEN.
7:13	IPLLEN	IP header length (including IP optional/extended headers) in the transmit buffer defined in DWords. In case of a tunnel packet, IPLLEN defines the most inner IP header length. If the IIPT flag is cleared, this field should be set to zero.
14:17	L4LEN	L4LEN is the L4 header length in the transmit buffer defined in DWords. In case of a tunnel packet, L4LEN defines the most inner L4 header length. L4LEN should obey the following rules: When the L4T field is set to 00b, L4LEN must be set to zero. Otherwise, it should be set to 8 / 12 for UDP / SCTP respectively and should be equal or larger than 20 for TCP.

L2 Tag 1 - L2TAG1 (Quad Word 1, 48:63)

A 16-bit Tag to be inserted into the packet if the IL2TAG1 flag is set. If IL2TAG1 is cleared, L2TAG1 should be set by software to zero.

Transmit Buffer Description - BUFF (Quad Word 0, bits 0:63; Quad Word 1, bits 34:47)

Table 8-21. Transmit Buffer

Bits	Name	Functionality
34:47	BSIZE	Buffer size in byte units from 1 byte up to 16KB minus 1
0:63	BADDR	Buffer address in byte granularity

8.4.2.1.2 NOP Descriptor

A NOP descriptor can be used if there is a software need to align descriptors. A NOP descriptor does not cause any activity other than processing of the descriptor.

NOP descriptors are implemented by Null setting of a context descriptor as follows: DTYP should be set to 0x1 (LAN Context Descriptor) and all other fields should be cleared. Note that NOP descriptors are permitted only between commands.



8.4.2.1.3 Transmit Descriptors Write Back Format

Quad Word	6																																	
	3																																0	
0	Reserved																																	
1	Reserved																														DTYP			
	6																																	
	3																															4	3	0

Descriptor Type - DTYP (Quad Word 1, bits 0:3)

The hardware indicates a completed descriptor by setting the DTYP field to a value of 0xF. The write back status is the same for all transmit descriptors used for LAN traffic, filter programming.

The hardware reports this status in the following two cases:

- For descriptors with the 'RS' bit set (Report Status) while the HEAD_WBEN flag in the transmit context is set to Descriptor Write Back mode. See rules for setting the 'RS' flag in the transmit descriptors in [Section 8.4.2.1.1](#).
- For the last descriptor of a command that was executed before an interrupt of the queue is initiated.

8.4.2.2 LAN Transmit Context Descriptors

8.4.2.2.1 Transmit Context Descriptor

A context descriptor may have additional setting options for a single packet or TSO defined by the data descriptor(s) that follows. Following the packet or TSO described in [Section 8.4.2.1](#), the context provided by this descriptor is lost.

See [Table 8.4.4](#) for field settings as a function of required offload.

Quad Word	6															4	4															3	3															2	2																
	3															8	7															2	1															4	3																0
0	rsv				L2TAG2 (STag / VEXT)								rsv				Tunneling Parameters																																																
1	MSS / TARGET_VSI				rsv				TSO Total Length								rsv				rsv				CMD				DTYP																																				
	6					5	4					4	4					3	2					1	1					1	1																																		
	3					0	9					8	7					0	9					8	7					1	0					4	3	0																											



Descriptor Type - DTYP (Quad Word 1, bits 0:3)

Command Field - CMD (Quad Word 1, bits 4:10)

Table 8-22. Transmit Command Field Descriptors

Bits	Name	Functionality
0	TSO	Transmit segmentation offload is activated when the TSO flag is set.
1	TSYN	1588 Time Stamp. When set, the hardware samples the packet transmission time in the PRTTSYN_TXTIME register. The TSYN flag is processed by the device only if TimeSync is enabled by the TSYNENA flag in the transmit queue context. Note that the TSYN flag is aimed only for a single send with no TSO.
2	IL2TAG2	Insert the L2 tag from the L2TAG2 field in this descriptor. For MAC in UDP or MAC in GRE tunneling, the L2TAG2 field is relevant to the outer L2 header. See IL2TAG_IL2H for VLAN tag insertion to the inner L2 header. The type of the Tag is defined by the L2TAG2INSERTID field in the VSI_L2TAGSTXVALID register of the VSI. The structure of this tag is defined by the matched GL_SWT_L2TAGxxx registers. See also IL2TAG1 in the data descriptor for 32 bit tag handling.
3	IL2TAG_IL2H	Insert VLAN to the inner L2 header from the L2TAG2 field in this descriptor. The VLAN is added after the header defined by L4TUNLEN. The IL2TAG2 and IL2TAG_IL2H are mutually exclusive and the software may set only one of them. The IL2TAG_IL2H can be set to '1' only for L4 tunneled packets with non-zero L4TUNLEN. Inserting the VLAN in the inner L2 header, the hardware also updates the IP total length field in the outer IP header. It is done for Single-Send as well as for TSO.
4:5	SWTCH	Switch Control Tag (Can be set to non-zero only by control VSI as programmed by the Allow destination override flag per VSI): 00b - No Switch control Tag. The packet is routed according to the hardware filters. 01b - Uplink packet. The packet is transmitted to the network bypassing hardware filters. 10b - Local packet. The packet is transmitted only to local VSI's according to the hardware filters. 11b - Target VSI. The packet is transmitted to a specific VSI defined in the MSS/VSI field in this descriptor. TSO is mutually exclusive with this option.
6	Reserved	Reserved

Segmentation Parameters / Switching Parameter (Quad Word 1, bits 30:47; bits 50:63)

Table 8-23. Switching Parameter Descriptors

Bits	Name	Functionality
30:47	TLEN	TSO Total Length. This field defines the L4 payload bytes that should be segmented. Note that the sum of all transmit buffer sizes of the TSO should match exactly the TLEN plus the TSO header size in host memory. If the TSO flag is cleared, the TLEN should be set by software to zero. If the TSO flag is set, the TLEN must be set by software to a larger value than zero.
50:63	MSS / TARGET_VSI	When the TSO flag is set, this field function as MSS. When the SWTCH field is set to 11b, this field function as TARGET_VSI. If the TSO flag is set then the SWTCH field should not be set to 11b and vice versa. If both the TSO flag is cleared and the SWTCH field is not equal to 11b then this field should be set to zero. When the TSO flag is set, the MSS field defines the Maximum Segment Size of the packet's payload in the TSO (excluding the L2, L3 and L4 headers). In case of tunneling the MSS relates to the inner payload. The MSS should not be set to a lower value than 64 or larger than 9668B. When the SWTCH field equals to 11b, it is the destination VSI of the packet. This option is valid only for control VSIs on which the Allow destination-override flag is set.



Tunneling Parameters (Quad Word 0, bits 0:23)

Table 8-24. Tunnelling Parameter Descriptors

Bits	Name	Functionality
0:1	EIPT	The External (outer) IP header type and its offload: 00 - no External IP header 01 - External IPv6 10 - External IPv4 with no checksum offload 11 - External IPv4 with checksum offload
2:8	EIPLLEN	External (outer) IP header length (including IP optional/extended headers) defined in DWords. When the packet has no outer IP header (EIPT equals to zero), this field must be set to zero.
9:10	L4TUNT	L4 Tunneling Type (Teredo / GRE header / VXLAN header) indication: 00b - No UDP / GRE tunneling (field must be set to zero if EIPT equals to zero) 01b - UDP tunneling header (Any UDP tunneling, VXLAN and Geneve) 10b - GRE tunneling header Else - reserved
11	RSV	Reserved zero
12:18	L4TUNLEN	L4 Tunneling Length (Teredo / GRE header / VXLAN header) defined in Words (field must be set to zero if L4TUNT equals to zero). For standard Teredo headers with no additional header payload it should be set to 4 which equals to 8 bytes. If the tunneling header includes proprietary content it should be included as well. For IP in GRE it should be set to the length of the GRE header. For MAC in GRE or MAC in UDP it should be set to the length of the GRE or UDP headers plus the inner MAC up to including its last Ethertype. Note that this field represents the length of data provided in host memory buffers. If the L4TUNT is cleared, this field must be set to zero.
19:22	DECTTL	Decrement TTL in the inner IP header by DECTTL and drop the packet if the original TTL was not greater than DECTTL. If the EIPT is cleared, this field must be set to zero.
23:24	Reserved	

L2 Tag 2 (Quad Word 0, bits 32:47)

Table 8-25. L2 Tag 2 Quad Word 0

Bits	Name	Functionality
32:47	L2TAG2	A 16 bit Tag to be inserted to the packet if the IL2TAG2 flag is set or if IL2TAG1 is set while L2 Tag 1 include 4 variable bytes or if the IL2TAG_IL2H flag is set. If the these conditions are not met, the L2TAG2 should be set by software to zero.

RSV Reserved bits that must be set to zero.

8.4.2.3 Filter Programming Descriptor

The Filter Programming Descriptor should be followed by data descriptor(s) as illustrated in [Figure 7-3](#).



Command Field - CMD (Quad Word 1, bits 4:19)

Table 8-28. Command Field Descriptors

Bits	Name	Functionality
0:1	PCMD	Filter Programming Command: 01b - Add Filter (*) 10b - Remove Filter Else - Reserved (*) If the filter entry does not exist, the filter is added. If the filter already exists, the filter parameters are updated.
2	RSV	Reserved
3:4	DEST	Matched packet destination control. 00b - Drop packet 01b - Direct packet to LAN queue defined by the QINDEX 10b - Direct packet to LAN while the queue is defined by other filters 11b - Reserved
5:6	RSV	Reserved (must be set to 00b)
7	CNT_ENA	Statistic Counter Enable flag (relevant only for FD filter). This flag enables packet counting by the statistic counter defined by the Statistic Counter Index in this command.
8	RSV	Reserved
9:10	FD_STATUS	The FD_STATUS flag enables status indication of the FD filter ID and extracting flexible bytes in the receive descriptor as follows: 00b - FD ID and Flexible bytes are not enabled 01b - FD ID is enabled 10b - FD ID and 4 Flexible bytes are enabled 11b - 8 Flexible bytes are enabled
11:15	RSV	Reserved
11:15	RSV	Reserved

8.4.3 LAN Transmit Queue (Ring)

The software prepares structures for transmission in system memory indicated to hardware by a list of consecutive descriptors. These descriptors are organized in a contiguous memory handled as a cyclic queue which is also called a descriptor ring. The X710-TM4 supports up to 1536 transmit queues allocated to PFs and VFs as described in [Section 8.2](#).

Transmit queues are defined by a set of parameters called the queue context. The main parameters are the queue pointers shown in [Figure 8-7](#). The queue context includes additional parameters that define the queue functionality as detailed in [Section 8.4.3.4](#). Part of these context parameters are kept in hardware (such as the Tail register, queue enable flag and interrupt related context). Most of the queue context parameters are stored in Function Private Memory (FPM), and fetched to an internal cache when required.

The software interface to the queue (for initialization, nominal operation, and queue disable flow) is described in [Section 8.4.3.1](#). The X710-TM4 has additional global parameters for the entire device or per function parameters that affect multiple queues. See [Section 8.3.3.1](#).

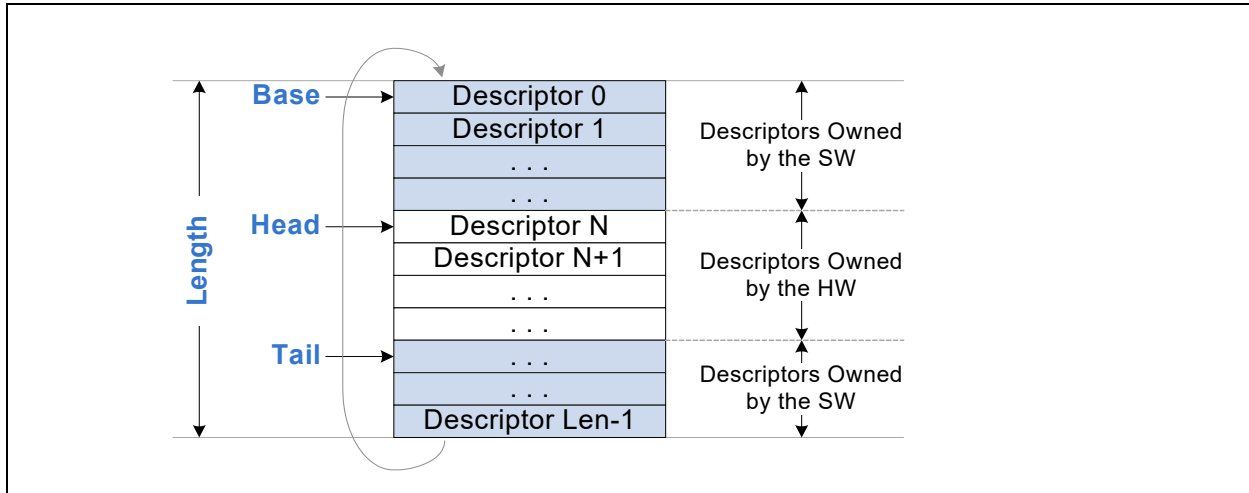


Figure 8-7. Transmit Descriptor Ring Structure

8.4.3.1 Transmit Queue Programming

Queue enable and disable flows are described in the following section and listed in Table 8-29.

Table 8-29. Transmit Queue Enable / Disable Flags

QTX_ENA[n] register		Transmit Queue State
QENA_REQ	QENA_STAT	
0	0	Queue is not enabled
1	0	Queue Enable request by the software.
1	1	Queue is enabled
0	1	Queue Disable request by the software.

8.4.3.1.1 Transmit Queue Enable Flow

Queue enable flow is executed by the PF software for PF queues as well as for the queues of its VFs.

- The function that owns the queue (PF or VF) allocates contiguous memory in its own memory space for the transmit ring.
- If the queue might have been disabled recently then the software should wait at least 50msec from the completion of the queue disable flow before proceeding to the next step.
- The software should clear the queue disable flag by setting the CLEAR_QDIS flag in the matched GLLAN_TXPRE_QDIS[n] register. The QINDX field in the GLLAN_TXPRE_QDIS[n] register should be set to the absolute queue index (equals to the queue index within the PF plus the FIRSTQ field in the PFLAN_QALLOC register of the PF).
- The software prepares the queue context in the FPM in PF memory space, clears the QTX_HEAD[n] register and then sets the QENA_REQ flag in the QTX_ENA[n] register, while 'n' is the queue index within PF space. Before that, the PF also defines the function that owns the queue in the PFVF_FUN; PF_INDX; VFVM_INDX parameters in the matched QTX_CTL[n] register. See several examples for



queue context settings in [Section 8.4.3.4.3](#). At pre-boot time, memory allocation might be tight and the FPM could exceed the budget.

Note: A mechanism to program the queue context directly to the device bypassing the FPM is described in [Section 8.4.3.1.4](#).

- As a response, the hardware sets the QENA_STAT flag in the QTX_ENA[n] register. The QENA_STAT follows the QENA_REQ almost instantly and not more than 10µsec after that.
- Once the QENA_STAT flag in the QTX_ENA[n] register is set, software can start using the queue.
- If the queue is targeted for a VF, PF software should program also the matched entry in the VFQ_TABLE. Then it is expected to inform the VF software that the queue is enabled. Note that the VFQ_TABLE is shared for the receive and transmit queues. So the PF software should also enable the receive queue before communicating this to the VF.

8.4.3.1.2 Transmit Queue Disable Flow

Queue disable flow is executed by PF software for PF queues as well as for the queues of its VFs.

- Remove the queue from the interrupt linked list as described in [Section 7.5.3.1.3](#).
- PF software clears the QENA_REQ flag in the QTX_ENA[n] register, while 'n' is the queue index within the PF space.
- Hardware schedules this event as it does for any other software update of the Tail register. Further updates of the Tail register are ignored by the hardware.
- Hardware posts a queue disable marker in the transmit pipe.
- Any preceding commands in the transmit pipe are processed while their buffers are fetched from host memory.
- Eventually, the queue disable marker gets to the end of the pipe. At this point it is guaranteed that the pipe does not contain any additional transmit commands that might require the host memory.
- The hardware invalidates the transmit queue from the internal cache and also clears the QENA_STAT flag in the QTX_ENA[n] register.
- Once the QENA_STAT flag in the QTX_ENA[n] register is cleared, software can release all memory structures of the queue. If the QENA_STAT flag in the QTX_ENA[n] register is not cleared within a reasonable time, it might be an indication for too long pause packets from the network. In such a case the software could follow the flow described in [Section 7.7.2.2.3](#).

Note: The progress of the queue disable marker following clearing the QENA_REQ flag is subject to scheduling in the transmit data path the same as any other packet. This means that processing latency might be greater if the traffic class is paused.



8.4.3.1.3 Fast Transmit Queue Disable Flow

Fast queue disable flow is executed by the hardware following a PFR, VFR or VMR resets. Once these reset are completed by the hardware, all transmit queues of the function or the VM are disabled. All queue contexts of the matched function or the VM are invalidated from the internal cache and their QENA_STAT flag in the matched QTX_ENA[n] register are cleared while the queue context is not updated in the FPM.

8.4.3.1.4 Direct Queue Context Programming

This section describes direct queue context programming, bypassing the FPM for pre-boot flow.

Context Programming Flow:

- Write the four PFCM_LANCTXDATA registers with a total of 16 bytes of context data.
- Write the PFCM_LANCTXCTL register identifying the queue and the specific sub-line to be programmed.
- Poll the CTX_DONE flag in the PFCM_LANCTXSTAT register till done is indicated.
- Loop back to the first step until all sub-lines of the queue context are programmed.

Context Invalidation Flow (this flow is not needed in nominal operation):

- Write the following fields in the PFCM_LANCTXCTL register: QUEUE_NUM and QUEUE_TYPE identify the queue and the OP_CODE fields should be set to Invalidate.
- Poll the CTX_DONE flag in the PFCM_LANCTXSTAT register until done is indicated.

8.4.3.1.5 Software Fast Path Programming

During normal operation, the function that owns the queue (PF or VF) accesses the hardware directly.

- Prepare transmit descriptors starting at the descriptor indicated by the TQTAIL pointer in the relevant QTX_TAIL register. Note that the software should update the TQTAIL at whole structure boundaries. For TSO, it is the whole TSO; for a single packet, it is the whole packet; and for filter programming, it is the end of the packet associated to the filter programming. Failing to do so might hang the activity of the queue.
- Software should never set the TQTAIL to a value above the descriptors owned by the hardware minus 1. Descriptors considered as owned by the hardware are those already indicated to the hardware but are not yet reported as completed.
- The hardware reports completed descriptors only for those indicated by an 'RS' bit in the CMD field in the descriptor. Completion indication is provided in one of two modes as programmed by the HEAD_WBEN parameter in the queue context as follows:
 - Descriptor Write Back: Hardware changes the value of the DTYP field in the completed descriptor to a 0xF (DTYP equals to 0xF is reserved for completed descriptor indication). It also sets the rest of the descriptors to all zeros.
 - Head Write Back: Hardware indicates the head pointer of the next descriptor that follows a completed descriptors (with active 'RS' bit). Hardware posts the 4 bytes of the head pointer at the address defined by the HEAD_WBADDR parameter in the queue context.
- Bump the TAIL to the last prepared descriptor plus one (the descriptor index to be added the next time).



8.4.3.2 Transmit During Link Down

In most usage models when the link becomes inactive, it is not required to preserve the transmit packets that were not sent. During link down, hardware continues to fetch transmit descriptors and data regardless of the link status. Packets directed to the network are discarded while packets directed to local VSI port are forwarded successfully.

Note that when the link becomes inactive, PFs on this port get a link status change interrupt. It is expected that the VFs get the link status change indication from their parent PF.

8.4.3.3 Transmit Arbitration Queue Set

Transmit queues are grouped into arbitration queue sets. The X710-TM4 supports 1024 LAN queue sets (indicated by the RDYList parameter in the transmit queue context). Queue arbitration within the same queue sets is a simple round robin scheme. Arbitration between queue sets is described in [Section 7.8.1.5](#). A queue context is fetched on demand (if not already in the cache) when it is scheduled. Then the hardware fetches the transmit descriptors (if not already in the cache) and then it fetches the transmit data.

8.4.3.4 Transmit Queue Context Parameters

This section describes the setting options of the LAN transmit queue parameters named as queue context. Part of the queue context parameters reside in dedicated hardware registers and part of the context is stored in host memory (FPM) while the active queues are kept in cache.

8.4.3.4.1 Transmit Queue Context in Hardware Registers

The queue context parameters that software accesses during fast past programming as well as its enablement, reside in hardware registers:

- Queue enablement flags in the QTX_ENA[n] registers while 'n' is the queue index of the PF. Queue enable and disable flow by the PF are explained in subsections of [Section 8.4.3.1](#).
- The TAIL pointer in the QTX_TAIL[n] registers while 'n' is the relative queue index of the function (PF or VF). The usage of the Tail register is detailed in [Section 8.4.3.1.5](#)
- Association of the queue to a function (PF and its VF or VM) is programmed by the QTX_CTL[n] registers while 'n' is the relative queue index of the PF. The QTX_CTL includes the following parameters that are programmed by the PF:
 - PF_INDIX is the absolute index of the PF (between 0 and 15)
 - PFVF_Q flag associate the queue to the PF, the EMP or to one of its VFs or its VMs.
 - VFVM_INDIX is the absolute index of the VF (between 0 and 127) if the queue belongs to a VF or the absolute index of the VSI (between 0 and 383) if the queue belongs to a VM. Otherwise, the VFVM_INDIX should be set to zero.

8.4.3.4.2 Transmit Queue Context in FPM

The transmit queue context parameters that are stored in host memory (FPM) are fetched for the active queues to the internal cache. These parameters are listed in [Table 8-30](#). The Queue context is structured as contiguous lines of 128 bits (16 bytes) each, as follows.



Table 8-30. LAN Tx Queue Context in the Private Host Memory

Alias	Width [bits]	LS bit	MS bit	Type	SW Init	Description
Line 0						
HEAD	13	0	12	Dynamic	0x0	Transmit Queue Head. An index updated by the hardware relative to the beginning of the queue that defines the next descriptor to be used. All descriptors starting by the HEAD up to (excluding) the TTAIL are owned by the hardware and the rest are owned by software (as shown in Figure 8.1). At idle time the HEAD equals to the TTAIL.
RSV	16	13	29	Dynamic	0x0	Reserved
New_Context	1	30	30	Dynamic	0x1	New Context Flag. Should be set to '1' by software at queue context programming.
RSV	1	31	31	N/A	0x0	Reserved
BASE	57	32	88	Static	BASE	Transmit Queue Base Address. Indicates the starting address of the descriptor queue defined in 128 Byte units.
TSYNENA	1	90	90	Static	TSYNENA	TimeSync Enable. If the TSYNENA flag is set, then setting the TSYN flag in the transmit context descriptor is processed by the hardware sampling the packet timestamp in the PRRTSYN_TXTIME register.
FDENA	1	91	91	Static	FDENA	Enable FD filter programming by this queue.
ALT_VLAN	1	92	92	Static	ALT_VLAN	Alternate VLAN tag selects. At '0' the PORT_TAG_ID tag in the VSI_TIR is disabled and at '1' the PORT_TAG_ID tag in the VSI_TAIR register is enabled.
RSV	3	93	95	N/A	0x0	Reserved
CPUID	8	96	103	Dynamic	CPUID	CPU Socket ID for TPH. The CPU socket ID is updated by the hardware
RSV	4	104	127	N/A	0x0	Reserved
Line 1						
THEAD_WB	13	0	12	Dynamic	0x0	Reflection of the reported HEAD at Head WB.
RSV	19	13	31	N/A	0x0	Reserved
HEAD_WBEN	1	32	32	Static	HEAD_WBEN	This flag selects between Head WB and transmit descriptor WB: 0b - Descriptor Write Back 1b - Head Write Back
QLEN	13	33	45	Static	QLEN	Transmit Queue Length (QLEN).- Defines the size of the descriptor queue in descriptors units from 8 descriptors (QLEN=0x8) up to 8K-32 descriptors (QLEN=0x1FE0). Q LEN restrictions: At smaller queue size than 32 descriptors the QLEN must be whole number of 8 descriptors. At larger size than 32 descriptors the QLEN must be whole number of 32 descriptors.
TPHRDesc	1	46	46	Static	TPHRDesc	Descriptor Read TPH Ena for descriptors fetch.
TPHRPacket	1	47	47	Static	TPHRPacket	Packet Content TPH Ena flag for the packet header and payload.
TPHWDesc	1	48	48	Static	TPHWDesc	Head WB / Descriptor Completion Status Write Back TPH Ena flag.
RSV	15	49	63	N/A	0x0	Reserved
HEAD_WBAD DR	64	64	127	Static	HEAD_WBAD DR	When the HEAD_WBEN is set to '1', this field defines the HEAD WB Address. Must be set to 0x0 if HEAD_WBEN is cleared.
Line 2						
RSV	128	0	127	N/A	0x0	Reserved
Line 3						
RSV	128	0	127	Tx_Desc	0x0	Reserved for internal parameters



Table 8-30. LAN Tx Queue Context in the Private Host Memory

Alias	Width [bits]	LS bit	MS bit	Type	SW Init	Description
Line 4						
RSV	128	0	127	Internal	0x0	Reserved for internal parameters
Line 5						
RSV	128	0	127	Internal	0x0	Reserved for internal parameters
Line 6						
RSV	128	0	127	Internal	0x0	Reserved for internal parameters
Line 7						
RSV	84	0	83	Internal	0x0	Reserved
RDYList	10	84	93	Static	RDYList	Transmit arbitration queue set. The RDYList index is absolute so it should be set to those RDYList allocated to the function.
RSV	34	94	127	Internal	0x0	Reserved

8.4.3.4.3 Examples for Transmit Queue Context Setting

Table 8-31. PF LAN Transmit Queue Context Example

Offset:	Even_L.0	Even_L.1	Even_L.2	Even_L.3	Odd_L.0	Odd_L.1	Odd_L.2	Odd_L.3
Line 0,1	40000000	001579A0	08000000	00000000	00000000	0001C200	00000000	00000000
Line 2,3	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
Line 4,5	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
Line 6,7	00000000	00000000	00000000	00000000	FFFFFFFF	480FFFFF	00000000	00000000

BASE = 0x001579A0 (0x0ABCD000)	HEAD_WBEN = 0
FCENA = 0	QLEN = 512 (0x200)
TSYNENA = 0	TPH = 111b (enable all TPHxxx flags)
FDENA = 1	HEAD_WBADDR = 0x00...0
ALT_VLAN = 0	RDYList = 0x80 (128)

8.4.4 Stateless Transmit Offloads

8.4.4.1 Insert Ethernet CRC bytes

See Section 7.2.1.1.

8.4.4.2 Transmit L3 and L4 Integrity Offload

The X710-TM4 can offload the following L3 and L4 integrity checks: IPv4 header(s) checksum for simple and tunneled packets, Inner TCP or UDP checksum and SCTP CRC integrity. Tunneling UDP headers and GRE header are not offloaded while the X710-TM4 leaves their checksum field as is. If a checksum is



required, software should provide it as well as the inner checksum value(s) that are required for the outer checksum. In order to request L3 and L4 Integrity Offloads, software should define the packet format and the required offload in the transmit descriptors as described in [Figure 8-8](#) and [Table 8-33](#).

Some rules for integrity offload are as follows:

- Offloads for inner headers to an IPv6 header with fragment extension header or fragmented IPv4 packets do not make sense. Note that this rule is not enforced by the X710-TM4.
- IPv6 support: The pseudo header for the L4 checksum takes into account the addresses in the IPv6 header ignoring the optional extension headers. Packets with Routing Header type 2 and Destination Options Header with Home Address option contain an alternative IP address in the extension header. Therefore the OS should not request checksum nor segmentation offload for packets with routing extension header type 2.
- IP Header Length Requirements (IPLLEN and EIPLLEN)
 - In case of non tunneled packet the IPLLEN defines the IP header length in DWord units and the IIPT defines the IP header type.
 - In case of tunneled packets, the IPLLEN defines the inner IP header length and the EIPLLEN defined by the outer (external) IP header length. The IIPT defines the inner IP header type and the EIPT defines the outer IP header type.
 - The Length field in the IP header is prepared by software in the packet buffers. This length field includes the payload of the IP header. In MAC tunneling, the inner MAC header (including optional VLAN tag) is part of the outer IP header payload. This optional VLAN tag can be embedded in the packet buffers or by using the IL2TAG_IL2H option in the transmit context descriptor. Regardless of these considerations, the optional VLAN tag should be taken into account in the length field of the outer IP header.
 - For IPv4 it should be at least 20 bytes (basic header size), and not more than 60 bytes.
 - For IPv6 it should be at least 40 bytes (basic header size), and up to the maximum size enabled by the parsing fields for basic IPv6 header and its extension headers.
- L4 Header Length Requirements (L4LEN):
 - For TCP, it should be at least 20 bytes (basic header size), and not more than 60 bytes.
 - For SCTP, it should be set to 12 (SCTP common header size).
 - For UDP, it should be set to 8 (UDP header size).

[Table 8-33](#) lists all supported packet formats and the processed integrity. The table uses the following notations:

- IP is a generic term for IPv4 header or IPv6 header. The IPv4 header can have IP option headers and the IPv6 header can have IPv6 extension headers.
- L4 is a generic term for UDP, TCP or SCTP headers.
- IP checksum is meaningful only for IPv4.
- Checksum is a generic term for UDP and TCP checksum as well as SCTP CRC integrity.

Offload Details:

- IPv4 checksum calculation (Inner or Single IPv4 only)
 - Software should set the IPv4 checksum to zero.
 - Hardware calculates the IPv4 header checksum starting at the beginning of the IPv4 header up to the end of the header.
- UDP and TCP checksum calculation (inner L4 header in case of UDP / GRE tunneling)
 - Software provides the pseudo IP header checksum in the L4 header.
 - Hardware calculates the L4 checksum starting at the beginning of the L4 offset up to the end of the packet which includes the pseudo header provided by software.



- SCTP CRC calculation (inner L4 header in case of UDP / GRE tunneling)
 - Software should set the CRC in the header to zero.
 - Hardware calculates the CRC according to SCTP standard starting at the beginning of the SCTP header up to the end of the packet.
- Tunneling UDP / GRE Header Length Requirements
 - For UDP / GRE tunneling, the header can be any value in 2 byte granularity from 8 bytes (bare Teredo UDP header) and up to 254 bytes (including optional network key and optional inner L2 header up to including the last Ethertype). Note that the header length includes only those bytes provided in the packet buffers in host memory.

Table 8-32 Pseudo Header Pre-loaded Values

Field	Single Send packet (No TSO)	Transmit Segmentation offload (TSO)
Inner/Single IP Length	IPv4: IP header +payload size IPv6: payload size only, including header extensions Should match the data size stored in host memory (Hardware will modify it upon offloads)	Don't Care (calculated by Hardware)
Inner/Single L4 (UDP/TCP) Length	Header size + Payload size Should match the data size that stored in host memories (Hardware will modify it upon offloads)	Don't Care (calculated by Hardware)
Inner/Single L4 CS field	IP header pseudo checksum, calculated on IP header fields: IPv4: Source addr +Destination addr + Protocol + L4 Length IPv6: Source addr +Destination addr + NextHeader + L4 Length Protocol or NextHeader values are: 0x11 for UDP and 0x6 for TCP L4 Length is the payload that follows the IP header. It includes the size of L4 header and data.	IP header pseudo checksum, calculated on IP header fields. Not including L4 Length: IPv4: Source addr +Destination addr + Protocol IPv6: Source addr +Destination addr + NextHeader Protocol or NextHeader values are: 0x11 for UDP and 0x6 for TCP

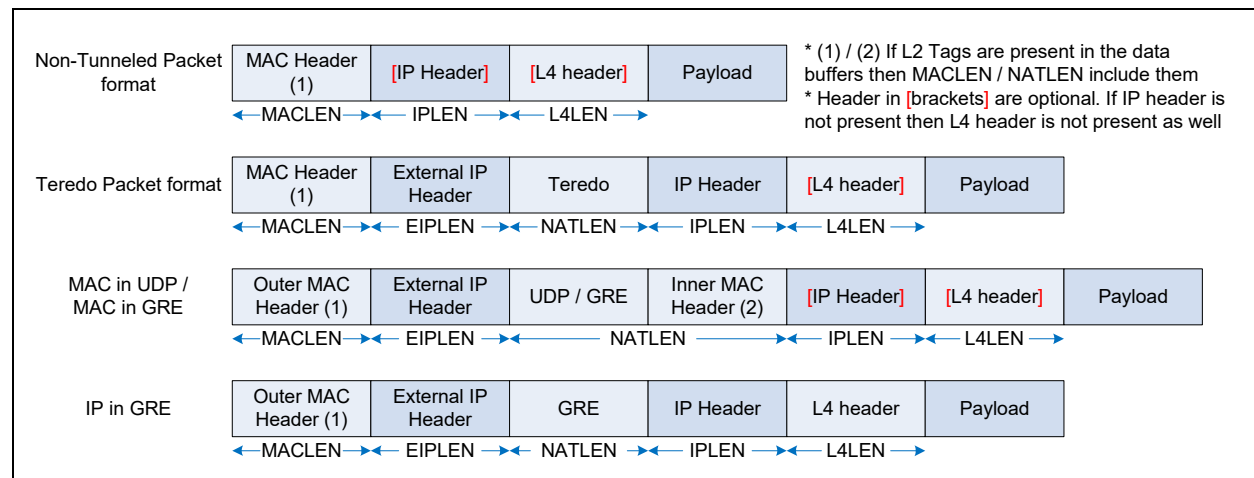


Figure 8-8. Transmit L3 and L4 Header Lengths (in Host Memory) for Integrity Offload



Table 8-33. Transmit Integrity Offload for Packet Types

Packet Type	Parsing hints and offload enablement in the transmit descriptors (1) (2)	Supported Transmit Checksum Offload
Fragmented IPv4 or IPv4 -> Unknown	IIPT = 10b or 11b L4T = 00b (unknown)	IPv4 checksum if IIPT = 11b.
Fragmented IPv6 or IPv6 -> Unknown	IIPT = 01b L4T = 00b (unknown)	None
IP -> L4	IIPT = 10b or 11b for IPv4 IIPT = 01b for IPv6 IPLLEN = the length of the IP header (including IP optional or extension headers). L4T = 11b, 01b, 10b (UDP, TCP, SCTP) L4LEN = L4 header length. EIPT = EIPLLEN = L4TUNT = 0 (no tunneling).	IPv4 checksum if IIPT = 11b: L4 checksum if L4LEN is meaningful (i.e., L4T <> 0).
IP -> (Fragmented IP or IP -> Unknown)	IIPT, IPLLEN, EIPT, EIPLLEN, L4TUNT = same as IP -> IP -> L4 L4T = 00b (unknown)	Inner IPv4 checksum if IIPT = 11b; Outer IPv4 checksum if EIPT = 11b; No L4 checksum
IP -> IP -> L4	IIPT = 10b or 11b for inner IPv4 IIPT = 01 for inner IPv6 IPLLEN = the length of the inner IP header (including IP optional or extension headers) L4T = 11b, 01b, 10b (UDP, TCP, SCTP) EIPT = 10b or 11b for outer IPv4; EIPT = 01 for outer IPv6 EIPLLEN = the length of the outer IP header (including IP optional or extension headers) L4TUNT = 00b (no L4 tunneling).	Inner IPv4 checksum if IIPT = 11b Outer IPv4 checksum if EIPT = 11b L4 checksum if L4LEN is meaningful (i.e., L4T <> 0)
IP -> Teredo UDP / GRE -> IP -> L4	IIPT = 10b or 11b for inner IPv4 IIPT = 01 for inner IPv6 IPLLEN = the length of the inner IP header (including IP optional or extension headers) L4T = 11b, 01b, 10b (UDP, TCP, SCTP) EIPT = 10b or 11b for outer IPv4 EIPT = 01 for outer IPv6 EIPLLEN = the length of the outer IP header (including IP optional or extension headers) L4TUNT = 01b for UDP/GRE tunneling L4TUNLEN = Tunneling header length	Same as IP -> IP -> L4
IP -> Teredo UDP / GRE -> (Fragmented IP or IP -> Unknown)	IIPT, IPLLEN, EIPT, EIPLLEN, L4TUNT, L4TUNLEN = same as IP -> IP -> L4 L4T = 00b (unknown)	Same as IP -> (Fragmented IP or IP -> Unknown)
IP -> GRE / UDP -> MAC (w/wo VLAN) -> IP -> L4	IIPT, IPLLEN, L4T, EIPT, EIPLLEN = same as IP -> IP -> L4 L4TUNT = 01b for UDP/GRE tunneling L4TUNLEN = UDP/GRE header length in the packet buffers up to excluding the inner IP header	Same as IP -> IP -> L4
IP -> GRE / UDP -> MAC (w/wo VLAN) -> (Fragmented IP or IP -> Unknown)	IIPT, IPLLEN, L4T, EIPT, EIPLLEN = same as IP -> (Fragmented IP or IP -> Unknown) L4TUNT, L4TUNLEN = same as IP -> GRE / UDP -> MAC (w/wo VLAN) -> IP -> L4	Same as IP -> (Fragmented IP or IP -> Unknown)
<p>Note 1. Common settings to all cases: When context descriptor is used, the TSO flag should be cleared.</p> <p>Note 2. When IP-IP or other supported tunneling is transmitted, a context descriptor must be used to provide the additional fields types and sizes. For non tunneled packets the context descriptor may be needed for other offloads than checksum. The values indicated in this table assumes that the context descriptor is used.</p>		

8.4.4.3 Transmit Segmentation Offload (also known as TSO or LSO)

Transmit Segmentation Offload (TSO) is also called Large Send offload (LSO). TSO enables the TCP/IP stack to pass to the network device a larger ULP datagram than the Maximum Transmit Unit Size (MTU). The X710-TM4 divides the large ULP datagram into multiple segments according to the MTU size as shown in the [Figure 8-9](#). The size of the ULP datagram supported for TSO can be as small as a single byte (obviously transmitted on a single segment) and up to 256KB (2^{18}) supporting TSO and Giant TSO.

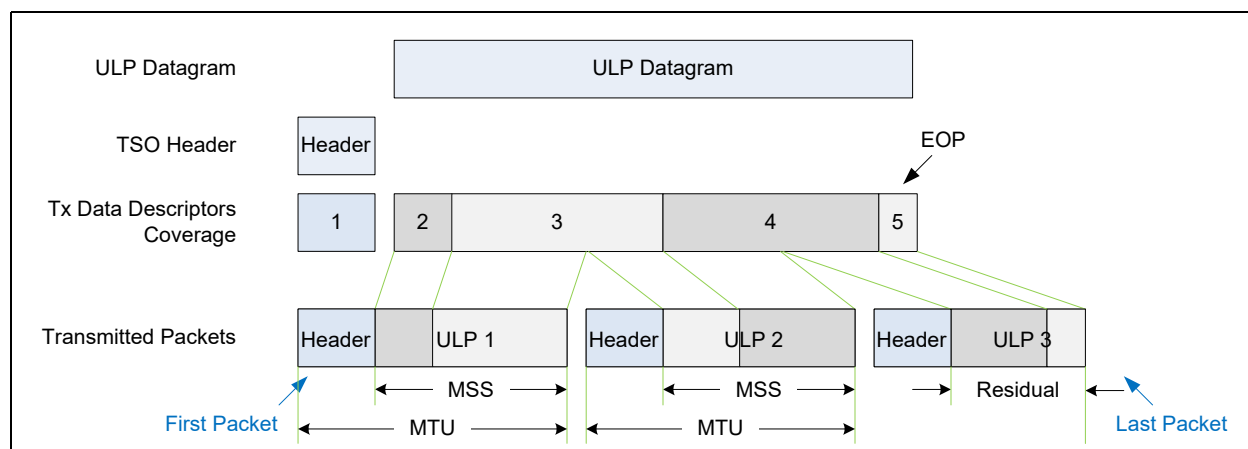


Figure 8-9. TSO Functionality (Example)

8.4.4.3.1 Frame Formats and Assumptions

The following packet formats are supported for TSO:

Note: For all the formats below, IP is IPv4 or IPv6 with/without Option/Extension headers. L4 is TCP.

- IP -> L4
- IP -> IP -> L4
- IP -> Teredo -> IP -> L4
- IP -> UDP Tunneling [Key] -> MAC -> [VLAN] -> IP -> L4
- IP -> GRE [Key] -> MAC -> [VLAN] -> IP -> L4
- IP -> GRE [Key] -> IP -> L4

SNAP packet formats are not supported for TSO. The following assumptions apply to the TCP segmentation implementation in the X710-TM4:

- TSO is activated by setting the TSO flag in the transmit context descriptor. Software should follow the data and context descriptor settings for integrity offload as listed in [Table 8-33](#).
- The TSO header (L2, L3 and L4) should be provided by a maximum three descriptors, while still allowed to mix header and data in the last header buffer. The maximum size of the TSO header is 512 bytes.
- The maximum size of a single TSO can be as large as 256 KB minus 1 (defined by the *TLEN* field in the transmit context descriptor).



- The *RS* bit in the data descriptor can be set only on the last data descriptor of the TSO (on which the EOP bit is set).
- It is assumed that software initializes the pseudo header checksum excluding the TCP length (as opposed to single send on which the pseudo header checksum includes the TCP length).

8.4.4.3.2 Transmit Segmentation Flow

The TSO flow includes the following steps:

- The protocol stack receives from an application a ULP datagram to be transmitted.
- The protocol stack calculates the number of packets required to be transmitted based on the MSS.
- The stack interfaces with the software device driver and passes the block down with the appropriate header information: Ethernet, IP header(s), optional tunneling headers and the L4 header.
- The stack interfaces with the software device driver and commands the driver to send the whole datagram. The software device driver sets up the interface to the hardware (via descriptors) for the segmentation.
- Hardware fetches the segmentation parameters as well as the data and header buffers description by the transmit context and data descriptors.
- Hardware fetches the header and data buffers and then transmits them by segments according to the TSO parameters.
- Dynamic fields set by the software:
 - For IPv4 the IP header checksum should be set to zero.
 - The Total Length field in the IP header(s) should be set to zero.
 - The IP ID of the first segment to be transmitted.
 - The pseudo header checksum of the TCP header should be calculated and placed as part of the packet data in the TCP or UDP checksum offset.
 - Initial value of the TCP flags (see later on for its value in the TSO packets).
- Dynamic fields in the IPv4 header(s) that are modified by hardware:
 - The Total Length field should reflect the IP payload size plus the IP header length. The L4 payload size is MSS for all packets but the last one which contains the rest of the data. So for the inner IP header (in case of tunneling), it is the L4 Payload + L4LEN + IPLEN. This is the same rule for the IP header in non tunneling case. For an outer IP header (in case of tunneling) it equals to: L4 Payload + L4LEN + IPLEN + optional L4TUNT length + EIPLEN.

- Note:** In case of MAC tunneling the length of the outer IP includes the inner L2 header including optional VLAN (updated by the hardware if the VLAN is inserted by the hardware).
- The Identification field (in the IP header in non tunneled packets or the inner IP header in tunneled packets) is taken from the TSO header in the first segment and then it is increased by one for each transmitted segment.
 - The Identification field (in the external IP header) is taken from the TSO header in the first segment and then it is either increased by one for each transmitted segment or remains constant depending on the EIP_NOINC setting in the transmit context descriptor.
 - The header checksum is calculated after the other parameters in the IP header are updated.
- Dynamic fields in the IPv6 header(s) that are modified by hardware:
 - The Payload Length should reflect the payload size. It is the MSS for all packets but the last one which contains the rest of the data. The Payload Length should reflect the IP payload size. So for the inner IP header (in case of tunneling) it is the L4 Payload + L4LEN + IP Extensions. This is the same rule for the IP header in non tunneling case. For an outer IP header (in case of tunneling) it equals to: L4 Payload + L4LEN + IPLEN + optional L4TUNT length + IP Extensions.



The IP Extensions length equals to IPLEN minus 40 or EIPLEN minus 40 for the inner or external IP headers respectively. Note that in case of MAC tunneling the length of the outer IP includes the inner L2 header including optional VLAN.

- Dynamic fields in the TCP header that are modified by hardware:
 - The Sequence number in the TCP header is taken from the TSO header in the first segment. It is then incremented by MSS for each transmitted segment.
 - The TCP flags are taken from the TSO header. The header is then masked (logic AND) by the TCPMSKF, TCPMSKM and TCPMSKL fields in the GLLAN_TSOMSK_F/M/L global registers. The TCPMSKF is used for the first segment of the TSO; TCPMSKL is used for the last segment of the TSO, and TCPMSKM is used for all other segments of the TSO. If the TSO is composed of a single segment, then it is processed the same as the last segment of a multiple segment TSO.
 - The TCP checksum is calculated starting from the initial value (of the pseudo header). The checksum includes the updated TCP header, TCP payload and the calculated TCP length (equals to the payload size plus the TCP header size).
- Dynamic fields in a tunneling UDP header (Teredo or MAC in UDP) that are modified by the hardware:
 - The UDP length reflects the size of the tunneling UDP payload plus 8 (which is the size of the UDP header).

8.4.4.3.3 Transmit Arbitration

The X710-TM4 arbitrates between transmit queues at packet boundaries while enabling each queue to transmit at least pre-defined quanta (as long as the queue is not empty). TSOs are not an exception to this rule. If the queue exhausts its quanta in the middle of a TSO, the X710-TM4 switches to the next queue in line at the end of the transmitted segment of the TSO. See the Transmit Scheduling chapter for a description of transmit arbitration.

8.4.4.3.4 Segmentation Indication to the Hardware

Software indicates a TCP segmentation using a transmit context descriptor just before the data descriptor with the following parameters:

- TSO flag is set, indicating a TSO is requested.
- Insert S-Tag or External VLAN can be set only by the PF (the same as a single send).
- Switch control fields must not be enabled for TSO.
- MSS should be set to the required size of the L4 payload on each segment (MTU minus the size of the headers).
 - MSS outside the range (256B - 9674B) are considered malicious. The respective queue is stopped and an interrupt is issued to the PF as described in [Section 7.6.2.2.1](#). The relevant event is Bad LSO MSS.
- TLEN is the total ULP datagram length.
- Other parameters in the data descriptor(s) and the context descriptor are defined the same as for a single send.

The data descriptors indicate the TSO header as well as the ULP datagram while following the Frame Formats and Assumptions described in [Section 8.4.4.3.1](#).



8.5 TimeSync (IEEE1588 and 802.1AS)

8.5.1 Overview

Measurement and control applications are increasingly using distributed system technologies such as network communication, local computing, and distributed objects. The 1588 standard enables accurate synchronization between clocks on distributed systems.

The X710-TM4 includes 1588 logic per LAN port; the logic is enabled by the TSYNENA flag in the PRRTSYN_CTL registers per port. The PFs can only access the 1588 registers of a port. All PFs on a given port can access the 1588 registers while only one PF is expected to control the logic. The PF ID that owns the 1588 logic of the port is reflected by the PF_ID in the PRRTSYN_CTL0 registers per port. All PFs are permitted to read the 1588 timer registers (on the port they belong to).

Software Note: The *PF_ID* field has no impact on the hardware; it is used only by the software. The field is expected to be loaded from NVM or set by management agent. During normal operation the PF software driver is not expected to change the field's setting.

The synchronized clock PRRTSYN_TIME is a 96-bit register (per LAN port). Out of it, the upper 64 bits are used for timestamp sampling and synchronized events described later in this section. Tune this register so that the upper 64 bits are defined in μ sec units. The register then can define the absolute time relative to PTP epoch which is January 1st 1970 00:00:00 International Atomic Time (TAI).

1588 Registers:

All 1588 registers are supported per port.

Many 1588 registers are 64 bits wide. The X710-TM4 provides a 32-bit interface. Therefore, accessing these registers requires multiple slave accesses.

- Reading any 64-bit register, software should access the LS register first and then the MS register. Specifically relating to dynamic registers: reading the LS register, the hardware latches internally the value of the MS register till it is fetched as well. Included in this category are: PRRTSYN_TIME; PRRTSYN_RXTIME; PRRTSYN_TXTIME; PRRTSYN_INC; PRRTSYN_TGT and PRRTSYN_EVNT registers.
- Writing to any 64-bit registers. software should write the LS register first and then the MS register. Only after writing the MS register is the value latched internally. Included in this category are: the PRRTSYN_TIME, PRRTSYN_INC, PRRTSYN_TGT, and PRRTSYN_EVNT registers. Writing to the PRRTSYN_TIME (*_L and *_H) registers, the device also clears the additional internal lowest 32 bits (usually used for sub nsec units). Note that this sub μ sec register is internal, and not exposed to the software.

8.5.2 Time Synchronization Flow

The operation of a 1588 logic is based on Precision Time Protocol (PTP). This protocol is composed of two stages: initialization and time synchronization. These stages are described below emphasizing hardware and software roles.



8.5.2.1 Initialization Phase

If enabled as a potential master (by a software setting), the software transmits sync packets that include the master's clock parameters periodically. Upon receipt of a sync packet, the software on any potential master compares the received clock parameters to its own parameters. If the received parameters are better, the software transitions to a slave state and stops sending sync packets.

While in slave state, the software selects a particular master. The software continuously compares the received Sync packet to its selected master. If the received Sync packet belongs to a different master with better clock parameters, the software switches to the new master. Eventually only one master (with the best clock parameters) remains active while all other nodes act as slaves listening to the single master.

Every node has a defined time-out interval. If no sync packets are received from the selected master each slave, software switches back to the initialization phase until a new master is chosen. Note that there are more than one option for the above flow while there are other flows which are based on static master setting. The node can be set statically to a master or slave modes. While in a slave mode, the node can be tuned statically to a specific master.

8.5.2.2 Time Synchronization Phase

There are two phases to the synchronization flow: (1) the slave calibrates its clock to the master and then (2) the master performs complete synchronization.

8.5.2.2.1 Clocks Calibration

The master sends SYNC packets periodically (about of 10 packets per second). These packets are followed by Follow_UP packets that indicate the transmission time. The slave captures the reception time of the SYNC packet; so it holds the packet transmission time at the master and its reception time at the slave. Receiving consecutive SYNC packets, the slave gets the delta T of the master and can calibrate its timer to get the same delta T. During this phase, the slave starts with an initial time calibration which can be used as the transmission delay between the master and the slave.

In order to minimize sampling inaccuracy, both master and slave sample the packets transmission and reception time at a location in the hardware that has as much as possible deterministic delay from the PHY interface.

Software Hardware Flow in the Master

Sending the SYNC packet by the master, software indicates this packet to the hardware by setting the TSYN flag in the transmit context descriptor. Setting this flag, the hardware samples its transmission time using the PRRTSYN_TXTIME register. After transmission, software should read the PRRTSYN_TXTIME register and sent the transmission time in a Follow_Up packet (according to 2-step flow). Note that software is responsible to read the PRRTSYN_TXTIME register before initiating another packet with an active TSYN flag.

Software Hardware Flow in the slave

The SYNC packet is received by the slave and its reception time is sampled using one of four PRRTSYN_RXTIME registers. The index of the register that sampled the reception time is reported in the *TSYNINDX* field in the receive descriptor. The PRRTSYN_RXTIME register holds the reception timestamp until the software reads it. It then released for sampling another packet reception.

8.5.2.2.2 Time Synchronization Phase

The complete synchronization scheme is illustrated below in [Figure 8-10](#). It relies on measured timestamp of Sync packets transmission and reception by the master and the slave. The scheme is based on two assumptions:

- The clocks at both nodes are almost identical (achieved in the first step).
- Transmission delays between the master to the slave and backward are symmetric.

The master’s software sends periodic Sync packets to each slave followed by a Follow_Up packet (as explained in the Software hardware flow in the master for the Clocks Calibration phase). The slave samples the TSYN packet reception time in one of the PRTTSYN_RXTIME registers.

The slave responds, sending a Delay_Request packet to the Master. The hardware in the slave samples its transmission time in the PRTTSYN_TXTIME register and the hardware in the master samples its reception time in one of the PRTTSYN_RXTIME registers. The software in the slave checks the Delay_Request packet transmission time (reading the PRTTSYN_TXTIME register) and could optionally send a Follow_Up packet that includes the captured transmission time. The master responds back with the Delay_Response packet reporting the reception time of the Delay_Response to the slave. At this point, the slave has all the following parameters (the notations below match the notations used in [Figure 8-10](#)):

- **T1**: Sync packet transmission time in the master (based on master clock)
- **T2**: Sync packet reception time in the slave (based on slave clock)
- **T3**: Delay_Request transmission time in the slave (based on slave clock)
- **T4**: Delay_Request reception time in the master (based on master clock)

The Slave can adjust its clock using the following equation:

- Slave Adjust Time = - $[(T2-T1) - (T4-T3)] / 2$

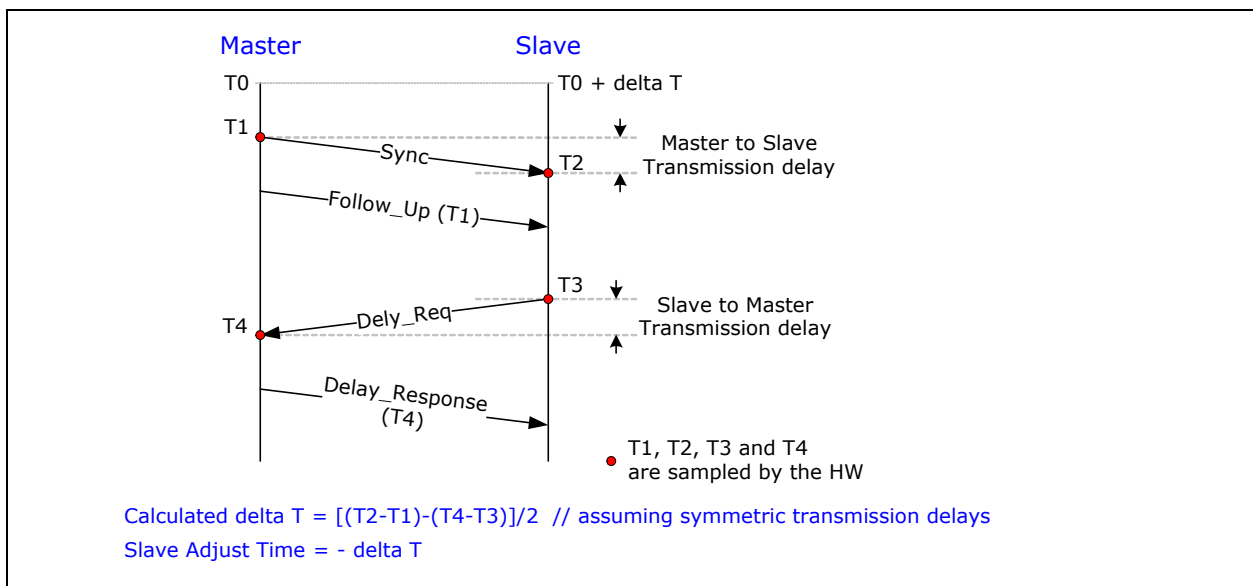


Figure 8-10. Sync Flow and Offset Calculation

PDelay Flow for Dynamic Master Selection



The master sends a Pdelay_Req packet to the slave and the slave response by sending back a Pdelay_Resp packet followed by a Pdelay_Resp_Follow_Up packet. The master uses these packets to calculate the link delay. This process is also used for fast recovery when the master is changed. The process is usually enabled when dynamic master selection is enabled. It can operate asynchronous to the *Time Synchronization* flow previously described. Figure 8-11 shows the Pdelay flow.

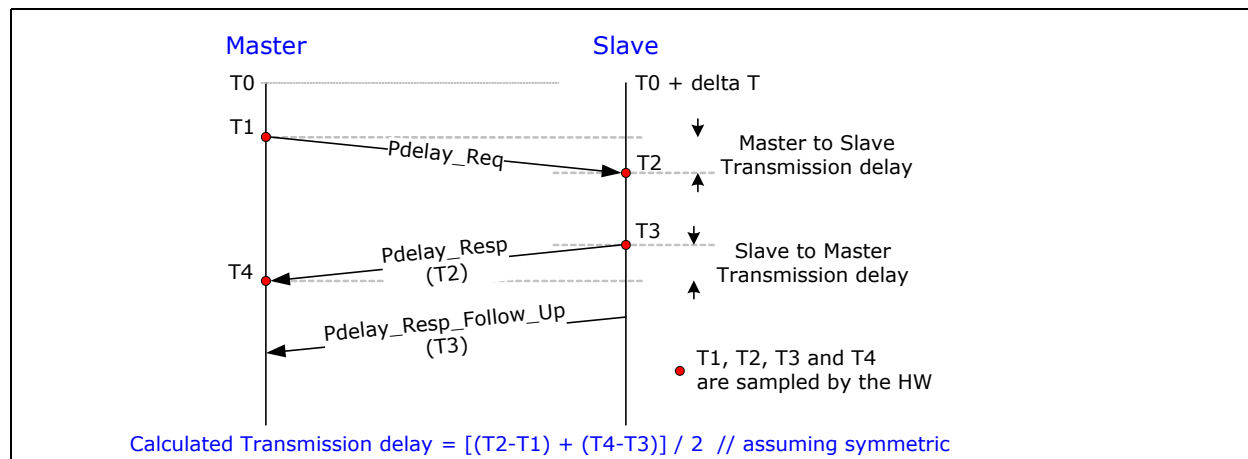


Figure 8-11. PDelay Flow

8.5.3 Timestamp Indication

Relevant packet transmission time (as Sync and Delay_Request) are sampled by the hardware at a deterministic as possible affinity to the PHY interface. The sampled time is taken at the beginning of the packet (packet size doesn't matter) as shown below.

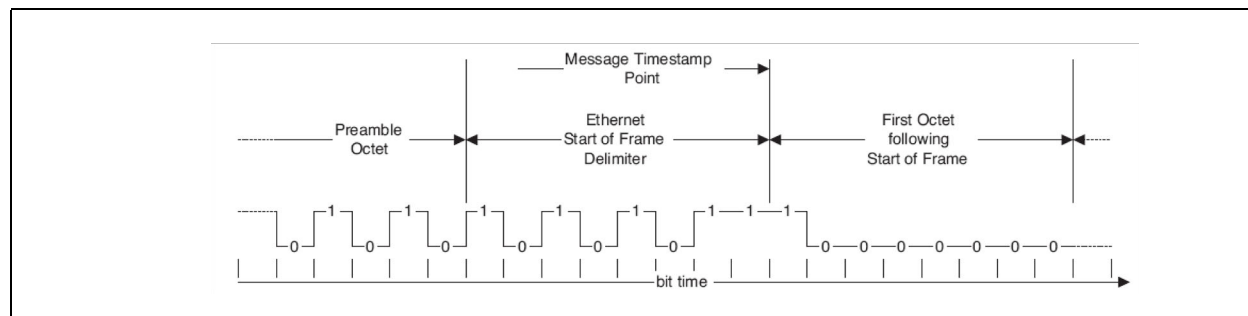


Figure 8-12. Time Stamp Point



8.5.3.1 Transmit Timestamp

Software indicates to the hardware the packets to be sampled by setting the TSYN bit in the transmit context descriptor. Hardware samples the transmission time of packets with an active TSYN bit, enabled for the queue by the TSYNENA flag in the transmit queue context. The transmission timestamp is sampled in the PRRTSYN_TXTIME register. Software is responsible for reading the PRRTSYN_TXTIME_L and PRRTSYN_TXTIME_H registers before sending another packet with active TSYN bit.



Note: In order to use the TSYN flag in the transmit descriptor, the TSYNENA flag should be set in the transmit queue context. Software should set the TSYNENA flag only for PF queues so VFs cannot interfere the Timesync functionality. Software should limit this enablement on a single PF per port which is assigned to control the 1588 logic.

8.5.3.2 Receive Timestamp

X710-TM4 identifies received PTP event packets as described in [Section 8.5.9](#). PTP packet reception time is sampled by one of 4 PRRTSYN_RXTIME[n] registers per port which is found free. Such packets are posted to LAN queues with an active 1588 TS Index (*TSYNINDEX*) field in the receive descriptor.

TSYNINDEX is the index of the PRRTSYN_RXTIME register that contains the packet reception timestamp. Once a PRRTSYN_RXTIME register samples a packet reception time, the register is locked until the software reads its value. When accessing this register the software should read first the PRRTSYN_RXTIME_L[n] register and then the PRRTSYN_RXTIME_H[n] register. Following read accesses to the PRRTSYN_RXTIME register, the lock is released and the register becomes free again.

Note that PRRTSYN_RXTIME registers are not accessible to any of the VFs (so they cannot interfere with functionality). The PRRTSYN_RXTIME are 64-bit registers that capture the upper 64 bits of the PRRTSYN_TIME register that belong to the LAN port on which the PTP packet was received (accessible to software by 2 x 32-bit accesses).

8.5.4 1588 Clock Registers

The clock driving the time sync elements is the MAC clock on the port. The MAC clock frequency depends on the link speed (see [Table 8-34](#)). Upon change in link speed the PRRTSYN_INC register (listed in [Table 8-34](#)) should change accordingly (this setting is the software responsibility).

Table 8-34. Proposed PRRTSYN_INC as a Function of MAC Clock Frequency

Link Speed	MAC Clock Frequency	Sampling Clock Precision	Recommended PRRTSYN_INC defining the 64 MS bits of PRRTSYN_TIME in nsec units
10 Gb/s	312.5 MHz	± 1.6 ns	13743895347 (0x333333333)
1 Gb/s	31.25 MHz	± 16 ns	137438953472 (0x2000000000)

The PRRTSYN_TIME is a 96-bit register holding the synchronized clock while its upper 64 bits are meaningful to the software accessible by 2 x 32-bit accesses. Each MAC clock, the PRRTSYN_TIME is incremented by PRRTSYN_INC. The lower 32 bits of the PRRTSYN_TIME register are internal, not accessible to software.

PRRTSYN_INC is a programmable 38-bit register accessible to software by 2 x 32-bit accesses (the upper 26 bits of this register are reserved and set to zero). When updating this register, software should set first the PRRTSYN_INC_L register and then the PRRTSYN_INC_H register. Hardware updates the internal value of the PRRTSYN_INC only after these two write cycles. Setting the PRRTSYN_INC by the recommended values in [Table 8-34](#), the upper 64 bits of the PRRTSYN_TIME are defined in ns units and the lower 32 bits of the PRRTSYN_TIME hold a fraction of a ns.

Software in slave node adjusts PRRTSYN_TIME in order to track the 1588 timer of the master using one of two methods:



- Direct write access to the upper 64 bits of the PRRTSYN_TIME register. This step could be useful for initial setting of the time during the Clocks Calibration phase.
- Fine adjustment during the 1588 Time Synchronization Phase using the PRRTSYN_ADJ register. The PRRTSYN_ADJ register is composed of a SIGN flag and *TSYNADJ* field that defines the absolute value of the adjustment. The *TSYNADJ* value is defined in the units of the PRRTSYN_TIME_L. The PRRTSYN_TIME is updated iteratively by the PRRTSYN_ADJ together with the nominal INC value on each MAC clock:

```
If (TSYNADJ = 0) then:
    PRRTSYN_TIME += PRRTSYN_INC
Else then:      // TSYNADJ is not equal to zero
    If positive adjust (SIGN=0) then: PRRTSYN_TIME += PRRTSYN_INC + 232
    Else : PRRTSYN_TIME += PRRTSYN_INC - 232
    TSYNADJ -= 1
```

8.5.5 Synchronized Auxiliary Events

X710-TM4 supports the following global auxiliary events for all ports:

- Synchronized events to global IO signals are described in [Section 8.5.5.1](#).
- Synchronized events to PCI slave access are described in [Section 8.5.6.1](#)

8.5.5.1 Auxiliary 1588 IO Signals

X710-TM4 supports 30 global GPIO signals that are controlled by 2 PRRTSYN_AUX[n] registers (n = 0,1). GPIO 'n' functionality is controlled by the GLGEN_GPIO_CTL[n] register (expected to be loaded from the NVM). GPIO's can be assigned to 1588 of a specific LAN port by the PRT_NUM and PIN_FUNC fields in the GLGEN_GPIO_CTL register. Setting the *PIN_FUNC* field to Timesync 0 or Timesync 1 assigns the GPIO to PRRTSYN_AUX[0] or PRRTSYN_AUX[1] respectively.

GPIO signals are set to input or output by the *PIN_DIR* field in the GLGEN_GPIO_CTL registers. When set to input signal the sampling event is sampled by the PRRTSYN_EVNT[n] registers ('n' is the matched index to the *PIN_FUNC* field). When set to output signal the output event time is defined by the PRRTSYN_TGT[n] register ('n' is the matched index to the *PIN_FUNC* field).

Several output modes of operation are described below. In any of these modes, the initial state of the GPIO output level can be controlled by setting the INSTNT and the OUTLVL flags in the PRRTSYN_AUX[n] register. All the output modes are enabled by the OUT_ENA flag in the PRRTSYN_AUX[n] register. There are also several input modes of operation controlled by the *EVNTLVL* field in the PRRTSYN_AUX[n] register.

Synchronized Level Output

This mode is selected by setting the PRRTSYN_AUX[n].OUT_ENA, setting the PRRTSYN_AUX[n].OUTMOD field to Output Level Mode and the event time is defined by the matched PRRTSYN_TGT[n] register. When the upper 64 bits of the PRRTSYN_TIME crosses the value of the PRRTSYN_TGT[n], the assigned GPIO transits to the requested level defined by the PRRTSYN_AUX[n].OUTLVL.



Synchronized Flipped Output Signal

Defined by the PRRTSYN_TGT[n] and PRRTSYN_AUX[n] registers similar to the Synchronized Level Output mode. The only difference is that for Flipped Output, the *OUTMOD* field should be set to Flipped Output Mode. In this case, each time the upper 64 bits of the PRRTSYN_TIME crosses the value of the PRRTSYN_TGT[n] the IO signal flips its output level.

Synchronized Output Pulse

Defined by the PRRTSYN_TGT[n] and PRRTSYN_AUX[n] registers similar to the Synchronized Level Output mode. The only difference is that the *OUTMOD* field should be set to Output Pulse Mode. In this case, each time the upper 64 bits of the PRRTSYN_TIME crosses the value of the PRRTSYN_TGT the GPIO signal flips its output state for $16 \times \text{PRRTSYN_AUX.PULSEW} + 1$ MAC clocks and then reverts back to its previous level.

Synchronized Clock Output

Defined by the PRRTSYN_TGT[n] and PRRTSYN_AUX[n] registers similar to the Synchronized Level Output mode with the following changes: The *OUTMOD* field should be set to Output Clock Mode and the matched PRRTSYN_CLKO[n] register should be set to 50% of the required output clock duration. Each time the upper 64 bits of the PRRTSYN_TIME crosses the value of the PRRTSYN_TGT the GPIO signal flips its output level. Then, the PRRTSYN_TGT register is reloaded by the hardware to PRRTSYN_TGT[n] plus PRRTSYN_CLKO[n] (while PRRTSYN_CLKO is padded by 32 MS bit zero's). Note that for proper operation the PRRTSYN_CLKO must be larger than PRRTSYN_INC + 2^{32} and for reasonable accuracy the PRRTSYN_CLKO should be significantly larger than PRRTSYN_INC + 2^{32} .

Sampling Input Event

X710-TM4 can capture the time on which an event is sensed on any of the 1588 auxiliary signals. The event time is captured by one of two matched 64 bit event time registers named PRRTSYN_EVNT[n]. The sampled input event type is defined by the *EVNTLVL* field in the PRRTSYN_AUX[n] register, equals to one of the following options: Disable; Rising Edge; Falling Edge; Any Transition. When the defined transition is sampled by the hardware (synchronized to the MAC clock), the upper 64 bits of the PRRTSYN_TIME are latched by the matched PRRTSYN_EVNT[n] register. Once the event is sampled, the PRRTSYN_EVNT register is locked, keeping the event time till the software reads it.

8.5.6 Synchronization with Host Timer

The time relationship between the CPU time and devices within the system is key element in many applications. Specifically, the relationship with the 1588 clock is critical. The following subsection describes mechanism to measure this relationship.

8.5.6.1 1588 PCI Slave Access

Following a write access to the PRRTSYN_AUX[n] while setting the *SAMPLE_TIME* flag, the hardware samples the PRRTSYN_TIME to the matched PRRTSYN_EVNT[n] register (the upper 64 bits of the PRRTSYN_TIME). This functionality can be useful to synchronize between a system timer accessible to the CPU and the PRRTSYN_TIME.



8.5.7 Interrupts

X710-TM4 can trigger an interrupt on any GPIO regardless of whether it is enabled for 1588 functionality (see [Section 7.5.2.3](#)). In addition, X710-TM4 can generate a 1588 interrupt for one of the following events (if enabled by the TIMESYNC flag in the PFINT_ICR0_ENA register):

- Packet transmission time is latched by the PRRTSYN_TXTIME registers while this interrupt is enabled by the TXTIME_INT_ENA flag in the PRRTSYN_CTL0 register.
- Input event is latched by one of the PRRTSYN_EVNT registers while this interrupt is enabled by the EVENT_INT_ENA flag in the PRRTSYN_CTL0 register.
- One of the time registers has expired while this interrupt is enabled by the TGT_INT_ENA flag in the PRRTSYN_CTL0 register (a target time register expires when PRRTSYN_TIME register is larger or equal than the target time register).

Regardless of interrupt enablement, the previous events are reported in the PRRTSYN_STAT_0 and PRRTSYN_STAT_1 registers. Following an interrupt assertion, software should read these registers to identify the cause. The registers are cleared on read, so they report an updated status since the last time they were read by the software.

8.5.8 1588 Initialization Flow

This section describes the PF software initialization flow required to activate 1588 logic.

- Check the PF_ID in the PRRTSYN_CTL0 register if the PF is expected to control 1588 logic.
- The 1588 CSRs are initialized by global reset which might not be triggered by a PCI reset. Therefore, PF software is required to initialize most of the 1588 registers listed in the TimeSync (IEEE 1588) Registers section:
 - Set PRRTSYN_CTL0 and PRRTSYN_CTL1 to the required 1588 packet type as well as required interrupt functionality.
 - Clear any optional residuals reported in the PRRTSYN_STAT register.
 - Read all receive timestamps, releasing them for nominal operation (PRRTSYN_RXTIME registers).
 - Set the PRRTSYN_AUX register as required for the AUX functionality.
 - The timer and the increment values are expected to be programmed as part of normal operation (PRRTSYN_TIME and PRRTSYN_INC registers).
- Assign a transmit queue for 1588 transmission and enable TSYNENA flag in its queue context.
- Optionally assign a receive queue. The EtherType Classification filter can be used to direct Layer 2 or UDP 1588 packets to a specific LAN receive queue.

8.5.9 PTP Packet Recognition

The time sync implementation supports both the 1588 Version 1 (V1) and Version 2 PTP frame formats (V2). The V1 structure is expected only as UDP payload over IPv4 while the V2 is expected over L2 with its EtherType or as a UDP payload over IPv4 or IPv6.

Note: X710-TM4 accepts the expected packet formats but does not enforce these UDP/IP rules.



The 802.1AS uses only the layer 2 V2 format. Note that PTP frame structure is not supported in the X710-TM4 for tunneled packets. The Layer 2 packet format and UDP packet format are shown in Figure 8-13. The PTP Message V1 and V2 formats are shown in Figure 8-36. The packet format supported by X710-TM4 for receive timestamp is controlled by the parameters shown in Figure 8-35.

Note: Correct UDP checksum is not a criteria for PTP packet recognition by the X710-TM4. Still, such a packet is reported to software with checksum error. The software can ignore the packet; but should read the register that carries the timestamp to release it for upcoming packets.

Note: the X710-TM4 does not recognize tunneled PTP packets.

Table 8-35. Receive Timestamp Control Parameters

Parameter	Register	Functionality
TSYNTP	PRTTSYN_CTL1	Controls which packets are sampled by the 1588 logic. It can be one of the following: <ul style="list-style-type: none"> L2 Version 2 packets while message type is defined by the PRTTSYN_CTL1 register UDP Version 1 packets while the UDP ports are defined by UDP_ENA and message type is defined by the PRTTSYN_CTL1 register L2 & UDP Version 2 packets while UDP ports and message type are defined as above L2 & UDP Version 2 event packets while UDP ports are defined by the UDP_ENA and the Message Type < 8
UDP_ENA	PRTTSYN_CTL1	Enable the UDP port numbers that are recognized as 1588 packets. It could be one of the following options: <ul style="list-style-type: none"> No UDP packet recognition UDP port number equals to 0x013F UDP port number equals to 0x0140 UDP port numbers equals to either 0x013F or 0x0140
V1MESSTYPE0, V1MESSTYPE1	PRTTSYN_CTL1	Controls the PTP V1 Message Type for sampled timestamp
V2MESSTYPE0, V2MESSTYPE1	PRTTSYN_CTL1	Controls the PTP V2 Message Type for sampled timestamp

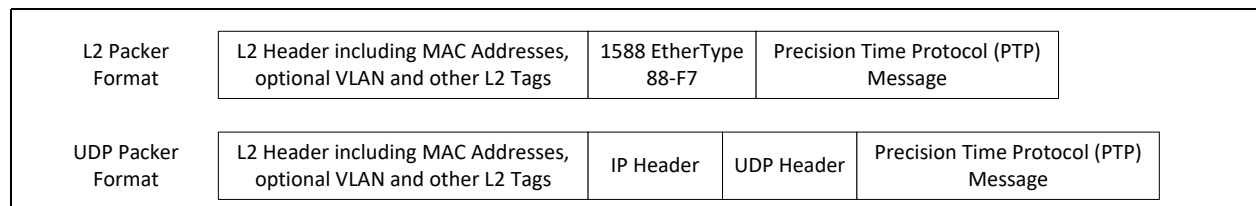


Figure 8-13. 1588 Packet Format

Table 8-36. PTP Header Format (V1 and V2 Formats)

Offset in Bytes	V1 Fields	V2 Fields	
		bits: 7 . . . 4	bits: 3 . . . 0
0	versionPTP	transportSpecific ¹	Message Type
1		Reserved	versionPTP
2 - 3	versionNetwork	messageLength	
4	Subdomain	SubdomainNumber	



Table 8-36. PTP Header Format (V1 and V2 Formats)

Offset in Bytes	V1 Fields	V2 Fields	
		bits: 7 . . . 4	bits: 3 . . . 0
5		Reserved	
6 - 7		flags	
8 - 15		Correction Field	
16 - 19		reserved	
20	Message Type	Source Port ID	
21	Source communication technology		
22 - 27	Sourceuuid		
28 - 29	sourceportid		
30 - 31	sequenceId	sequenceId	
32	control	control	
33	reserved	logMessagePeriod	
34 - 35	flags	n/a	

1. Should all be zero.

Table 8-37. PTP V2 and V1 Message Types

Message Type	Message Class	V2 Message Type	V1 Message Type
Sync	Event	0x0	0x00
Delay_Req	Event	0x1	0x01
Pdelay_Req	Event	0x2	N/A
Pdeley_Resp	Event	0x3	N/A
Reserved	-	0x4 - 0x7	N/A
Follow_Up	General	0x8	0x02
Delay_Resp	General	0x9	0x03
Pdelay_Resp_Follow_Up	General	0xA	N/A
Announce	General	0xB	N/A
Signaling	General	0xC	N/A
Management	General	0xD	0x04
Reserved	-	0xE, 0xF	0x05 - 0xFF

8.6 Software Timer

X710-TM4 includes a 32-bit counter in GLVFGEN_TIMER register which is based on a free running 1 μs clock. The counter is cleared by Core Reset (CORER) and increments after being cleared. The timer wraps around in about 70 minutes.



9.0 System Manageability

Network management is an important requirement in today's networked computer environment.

Software-based management applications provide the ability to administer systems while the operating system is functioning in a normal power state (not in a pre-boot state or powered-down state). The Intel® Out of Band Management fills the management void that exists when the operating system is not running or fully functional. This is accomplished by providing mechanisms by which manageability network traffic can be routed to and from a Management Controller (MC).

This section describes the supported management interfaces and hardware configurations for platform system management. It describes the interfaces to an external MC, the partitioning of platform manageability among system components, and the functionality provided by the X710-TM4 in each platform configuration.

9.1 Pass-through (PT) Functionality

PT is the term used when referring to the process of sending and receiving Ethernet traffic over the sideband interface. The X710-TM4 has the ability to route Ethernet traffic to the host operating system as well as the ability to send Ethernet traffic over the sideband interface to an external MC. See [Figure 9-1](#).

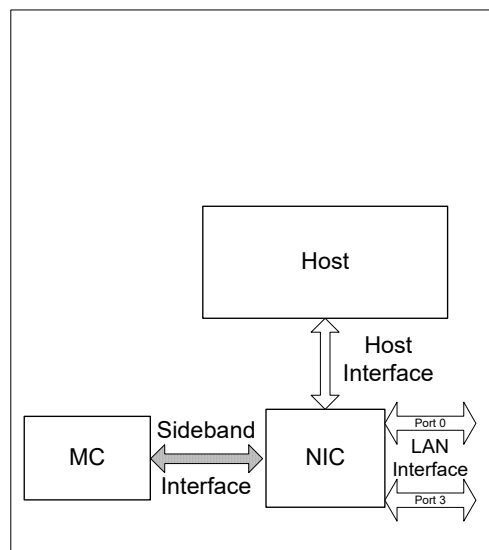


Figure 9-1. Sideband Interface



The sideband interface provides a mechanism by which the X710-TM4 can be shared between the host and the MC. By providing this sideband interface, the MC can communicate with the LAN without requiring a dedicated Ethernet controller. The X710-TM4 supports three sideband interfaces:

- SMBus (MCTP)
- NC-SI
- PCIe (together with MCTP) - when the system is up

Note: The usable bandwidth for either direction is up to 1 Mb/s when using SMBus and 100 Mb/s for the NC-SI interface. When working over PCIe, the bandwidth is limited only by the PCIe bandwidth and the the X710-TM4 processing capabilities and can sustain any network bandwidth. The X710-TM4 should support MCTP over PCIe pass-through traffic at a rate of up to 1 Gb/s. The maximum packet size supported for traffic received from the LAN to the MC is 1518 bytes and an additional S-tag or VLAN tags. For traffic from the MC to the LAN the maximum supported packet size is 1536 bytes including all tags.

Note: In MCTP mode, the PCIe and SMBus interface can receive MCTP commands in parallel. For example, the MCTP enumeration process can be done both over SMBus and over PCIe. However, only one of the interfaces can receive NC-SI commands or pass-through traffic.

9.1.1 Supported Topologies

The X710-TM4 can support up to two connections to management controllers in some topologies. The following connections are available:

- Connection via legacy SMBus (See [Section 9.5](#))
- Connection via NC-SI over RBT (RMII) (See [Section 9.6](#))
- Connection via NC-SI over MCTP. This connection can be over SMBus, PCIe or both. This connection can be used either for pass through or for control only (See [Section 9.6](#))
- Two connections — A connection via NC-SI over RBT for pass-through traffic and a connection via MCTP used only for control traffic (See [Section 9.6](#))

The channel used for pass through is defined in the *Redirection Sideband Interface* field and the channel used for control is defined in the *Control Interface* field, both in the *Common Manageability Parameters* NVM word (see [Section 6.2.36.2](#)) and are common to all the ports in the device.

9.1.2 Pass Through Packet Routing

When an Ethernet packet reaches the X710-TM4, it is examined and compared to a number of configurable filters. These filters are configurable by the MC and include, but not limited to, filtering on:

- MAC address
- IP address
- UDP/IP ports
- VLAN tags
- Ethertype

If the incoming packet matches any of the configured filters, it is passed to the MC. Otherwise, it is not passed.



The packet filtering process is described in [Section 9.3](#).

9.2 Components of the Sideband Interface

There are two components to a sideband interface:

- Physical layer
- Logical layer

9.2.1 Physical Layer

This is the electrical connection between the X710-TM4 and the MC.

9.2.1.1 SMBus

The SMBus physical layer is defined by the SMBus specification. The interface is made up of two connections: data and clock. There is also an optional third connection: the alert line. This line is used by the X710-TM4 to notify the MC that there is data available for reading in legacy SMBus mode. Refer to the SMBus specification for details.

The SMBus can run at two speeds: 100 KHz (standard SMBus) or 400 KHz (I²C fast mode). The speed used while the X710-TM4 is the master of the bus is selected by the *SMBus Connection Speed* field in the SMBus Notification Timeout and Flags NVM word. When acting as a slave, the X710-TM4 can receive transaction with a clock running at up to 400 KHz.

9.2.1.1.1 PEC Support

SMBus transactions can be protected by using Packet Error Code (PEC). Packet error checking, when applicable, is implemented by appending a PEC byte at the end of each message transfer. The PEC byte is a CRC8 calculation on all the message bytes.

PEC is added in transmit and expected in receive for the following SMBus packets:

- ARP packets
- MCTP over SMBus transactions.

For ARA cycles and legacy SMBus transactions, a PEC is not expected.

The following table lists the behavior of the X710-TM4 in each PEC configured mode for transactions directly handled by hardware after receiving packets with or without PEC.



Table 9-1. SMBus PEC Modes¹

SMBus Transaction (Relative to the X710-TM4)	X710-TM4 PEC Mode	Target PEC Mode	
		PEC Enabled	PEC Disabled
Master Write ²	Enabled	(A) Target ACKs the PEC byte	(A) Target NACKs the PEC byte
Master Write ²	Disabled	(A) Target receives stop before expected PEC byte	(A) PEC byte is not expected
Slave Write ³	Enabled	(A) Target ACKs last data byte; PEC byte is NACKed	(A) Target NACKs last data byte; No PEC byte is written by the slave
Slave Write ³	Disabled	(A) Target ACKs last data byte; PEC byte is 0xFF	(A) Target NACKs last data byte and generates stop afterwards
Slave Read ⁴	Enabled	(A) Target sends the PEC byte; PEC byte is ACKed by the slave	(A) Target does not send PEC byte and generates stop afterwards
Slave Read ⁴	Disabled	(R) Target sends the PEC byte; PEC byte is NACKed by the slave	(A) Target does not send PEC byte and generates stop afterwards

1. (A) - Accept transaction; (R) - Reject transaction.
2. Used in Legacy SMBus writes commands (direct receive) and in MCTP over SMBus (transmitted transactions).
3. Used in Legacy SMBus Read commands.
4. Used in Legacy SMBus mode (alert/async-notify) and in MCTP over SMBus (received transactions).

Note: In both SMBus ARP and MCTP, the specification indicates that PEC must be used. However, if PEC is not used by the master, the transaction is still accepted and processed by the X710-TM4.

The PEC behavior is controlled by the SMBus transaction PEC bit in the SMBus Notification Timeout and Flags NVM word. If this bit is set, PEC is added for master SMBus write transactions. A PEC is added to slave read transactions and can be received in slave write transaction. If this bit is cleared, PEC is not added to master write or slave read transactions, a slave write transaction with PEC is dropped. This bit is set.

9.2.1.2 NC-SI over RBT

The X710-TM4 uses the DMTF standard RBT sideband interface as defined in DMTF DSP0222. This interface consists of seven lines for transmission and reception of Ethernet packets and two optional lines for arbitration among more than one physical network controller.

The RBT physical layer of NC-SI is very similar to the RMII interface, although not an exact duplicate. Refer to the NC-SI specification for details of the differences.

9.2.1.2.1 Electrical Characteristics

The X710-TM4 complies with the electrical characteristics defined in the NC-SI 1.0.1 specification.

The X710-TM4's NC-SI behavior is configured at power-up in the following manner:

- The *Multi-Drop NC-SI* NVM bit (Section 6.2.36.3) defines the NC-SI topology (point-to-point or multi-drop; the default is point-to-point).

The X710-TM4 dynamically drives its NC-SI output signals (NC-SI_DV and NC-SI_RX) as required by the sideband protocol:

- At power-up, the X710-TM4 floats the NC-SI outputs.



- If the X710-TM4 operates in point-to-point mode, it starts driving the NC-SI outputs some time following power-up.
- If the X710-TM4 operates in a multi-drop mode, it drives the NC-SI outputs as configured by the BMC.

9.2.1.3 PCIe Vendor Defined Messages (VDMs)

The X710-TM4 uses VDMs over PCIe defined in the DMTF MCTP specification to convey pass-through traffic or NC-SI control traffic. See [Section 3.1](#) for details of the PCIe interface.

9.2.2 Logical Layer

9.2.2.1 SMBus

9.2.2.1.1 SMBus Transactions

This section gives a brief overview of the SMBus protocol. Following is an example for a format of a typical SMBus transaction.

Table 9-2. Typical SMBus Transaction

1	7	1	1	8	1	8	1	1
S	Slave Address	Wr	A	Command	A	PEC	A	P
	1100 001	0	0	0000 0010	0	[Data Dependent]	0	

The top row of the table identifies the bit length of the field in a decimal bit count. The middle row (bordered) identifies the name of the fields used in the transaction. The last row appears only with some transactions, and lists the value expected for the corresponding field. This value can be either hexadecimal or binary.

The SMBus controller is a master for some transactions and a slave for others. The differences are identified in this document.

Shorthand field names are listed in [Table 9-3](#) and are fully defined in the SMBus specification.

Table 9-3. Shorthand Field Names

Field Name	Definition
S	SMBus START Symbol.
P	SMBus STOP Symbol.
PEC	Packet Error Code.
A	ACK (Acknowledge).



Table 9-3. Shorthand Field Names

Field Name	Definition
N	NACK (Not Acknowledge).
Rd	Read Operation (Read Value = 1b).
Wr	Write Operation (Write Value = 0b).

9.2.2.1.2 SMBus Addressing

In legacy SMBus mode, the SMBus is presented as up to four SMBus devices on the SMBus (four addresses). All PT functionality is duplicated on the SMBus address, where each SMBus address is connected to a different LAN port. Note that it is not permitted to configure multiple ports to the same SMBus address. When a LAN function is disabled, the corresponding SMBus address is not presented to the MC.

In MCTP mode, a single SMBus channel is exposed.

SMBus addresses (enabled from the NVM) can be re-assigned using the SMBus ARP protocol.

In addition to the SMBus address values, all parameters of the SMBus (SMBus channel selection, address mode, and address enable) can be set only through NVM configuration. Note that the NVM is read at the X710-TM4's power up and resets.

9.2.2.1.3 SMBus ARP Functionality

The X710-TM4 supports the SMBus ARP protocol as defined in the SMBus 2.0 specification. The X710-TM4 is a persistent slave address device so its SMBus address is valid after power-up and loaded from the NVM. The X710-TM4 supports all SMBus ARP commands defined in the SMBus specification both general and directed.

SMBus ARP capability can be disabled through the NVM.

9.2.2.1.3.1 SMBus ARP Flow

SMBus ARP flow is based on the status of two flags:

- Address Valid (AV) – This flag is set when the X710-TM4 has a valid SMBus address.
- Address Resolved (AR) – This flag is set when the X710-TM4 SMBus address is resolved (SMBus address was assigned by the SMBus ARP process).

These flags are internal X710-TM4 flags and are not exposed to external SMBus devices.

Since the X710-TM4 is a Persistent SMBus Address (PSA) device, the AV flag is always set, while the AR flag is cleared after power up until the SMBus ARP process completes. Since AV is always set, the X710-TM4 always has a valid SMBus address.

When the SMBus master needs to start an SMBus ARP process, it resets (in terms of ARP functionality) all devices on SMBus by issuing either Prepare to ARP or Reset Device commands. When the X710-TM4 accepts one of these commands, it clears its AR flag (if set from previous SMBus ARP process), but not its AV flag (the current SMBus address remains valid until the end of the SMBus ARP process).



Clearing the AR flag means that the X710-TM4 responds to SMBus ARP transactions that are issued by the master. The SMBus master issues a Get UDID command (general or directed) to identify the devices on the SMBus. The X710-TM4 always responds to the Directed command and to the General command only if its AR flag is not set.

After the Get UDID, The master assigns the X710-TM4 SMBus address by issuing an Assign Address command. The X710-TM4 checks whether the UDID matches its own UDID and if it matches, it switches its SMBus address to the address assigned by the command (byte 17). After accepting the Assign Address command, the AR flag is set and from this point (as long as the AR flag is set), the X710-TM4 does not respond to the Get UDID General command. Note that all other commands are processed even if the AR flag is set. The X710-TM4 stores the SMBus address that was assigned in the SMBus ARP process in the NVM, so at the next power up, it returns to its assigned SMBus address.

Figure 9-2 shows X710-TM4 SMBus ARP flow.

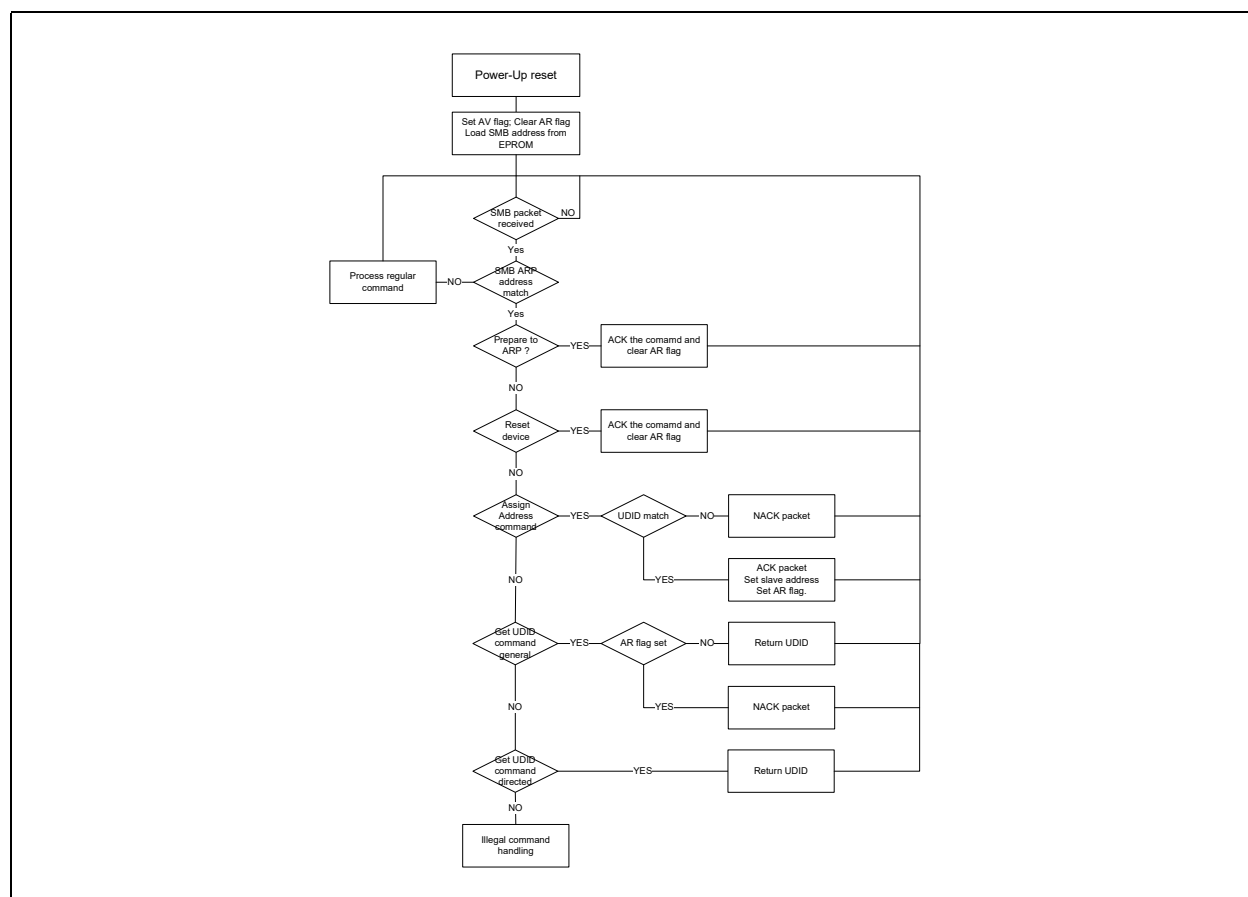


Figure 9-2. SMBus ARP Flow

9.2.2.1.3.2 SMBus ARP UDID Content

The UDID provides a mechanism to isolate each device for the purpose of address assignment. Each device has a unique identifier. The 128-bit number is comprised of the following fields:



Table 9-4. UDID

1 Byte	1 Byte	2 Bytes	2 Bytes	2 Bytes	2 Bytes	2 Bytes	4 Bytes
Device Capabilities	Version/Revision	Vendor ID	Device ID	Interface	Subsystem Vendor ID	Subsystem Device ID	Vendor Specific ID
See notes that follow	See notes that follow	0x8086	0x15FF	0x0004/ 0x0024	0x0000	0x0000	See notes that follow
MSB							LSB

Where:

- Vendor ID: The device manufacturer’s ID as assigned by the SBS Implementers’ Forum or the PCI SIG. Constant value: 0x8086.
- Device ID: The device ID as assigned by the device manufacturer (identified by the Vendor ID field). Constant value: 0x15FF.
- Interface: Identifies the protocol layer interfaces supported over the SMBus connection by the device. Bits 3:0 = 0x4 indicates SMBus Version 2.0. Bit 5 (ASF bit) = 1 in MCTP mode.
- Subsystem Fields: These fields are not supported and return zeros.

Device Capabilities: Dynamic and Persistent Address, PEC Support bit:

7	6	5	4	3	2	1	0
Address Type		Reserved (0)	Reserved (0)	Reserved (0)	Reserved (0)	Reserved (0)	PEC Supported
0b	1b	0b	0b	0b	0b	0b	0/1b ¹
MSB							LSB

1. The value is set according to the SMBus transaction PEC bit in the NVM.

Version/Revision: UDID Version 1, Silicon Revision:

7	6	5	4	3	2	1	0
Reserved (0)	Reserved (0)	UDID Version			Silicon Revision ID		
0b	0b	001b			See the following table		
MSB							LSB



Silicon Revision ID:

Silicon Version	Revision ID
A0	000b
B0	001b
B1	010b

Vendor Specific ID: Four LSB bytes of the device Ethernet MAC address of the relevant port. The port Ethernet address is taken from the *PRTGL_SAL* registers of the relevant ports. Note that in the X710-TM4 there are up to four MAC addresses (one for each port).

1 Byte	1 Byte	1 Byte	1 Byte
MAC Address, Byte 3	MAC Address, Byte 2	MAC Address, Byte 1	MAC Address, Byte 0
MSB			LSB

9.2.2.1.3.3 SMBus ARP and Multi-port

The X710-TM4 responds as four SMBus devices having four sets of AR/AV flags (one for each port). The X710-TM4 responds four times to the SMBus ARP master, once each for each port. All SMBus addresses are taken from the SMBus ARP address word of the NVM.

Note that the Unique Device Identifier (UDID) is different for each of the four ports in the version ID field, which represents the MAC address and is different for the four ports. The X710-TM4 first responds as port 0, and only when an address is assigned, then start responding as other ports to the Get UDID command.

9.2.2.1.4 Concurrent SMBus Transactions

The SMBus interface is single threaded. Thus, concurrent SMBus transactions are not permitted. Once a transaction starts, it must complete before an additional transaction is initiated.

A transaction is defined as:

- All the SMBus commands used to receive a packet.
- All the SMBus commands used to send a packet.
- The read and write SMBus commands used as part of read parameters described in [Section 9.5.10.3](#).
- The single write SMBus commands described in [Section 9.5.10.1](#).

9.2.2.2 Legacy SMBus

The protocol layer for SMBus consists of commands the MC issues to configure filtering for X710-TM4 management traffic and the reading and writing of Ethernet frames over the SMBus interface. There is no industry standard protocol for sideband traffic over SMBus. The protocol layer for SMBus on the X710-TM4 is Intel proprietary. The legacy SMBus protocol is described in [Section 9.5](#).



9.2.2.3 NC-SI

The DMTF defines the protocol layer for the NC-SI interface. NC-SI compliant devices are required to implement a minimum set of commands. The specification also provides a mechanism for vendors to add additional capabilities through the use of OEM commands. Intel OEM NC-SI commands for the X710-TM4 are discussed in [Section 9.6.4](#). For information on base NC-SI commands, see the NC-SI specification.

NC-SI traffic can run on top of three different Physical layers:

1. NC-SI physical layer as described in [Section 9.2.1.2](#).
2. MCTP over PCIe VDM. This protocol enables control and pass-through traffic over PCIe of a NIC or a LOM device. The NC-SI over MCTP protocol is slightly different than the standard NC-SI as it includes additional NC-SI commands. This mode is usually paired with an MCTP over SMBus, where this mode is used in S0 states and the SMBus interface is used in Sx state. The MCTP protocol and the differences from standard NC-SI is described in [Section 9.6](#).
3. MCTP over SMBus. As previously described, this layer is paired with the MCTP over PCIe to support Sx modes.

The X710-TM4 exposes one NC-SI package with four channels, one per port. The X710-TM4 implements a type C NC-SI interface (single package, common bus buffers and shared RX queue) as described in section 5.2 of the NC-SI specification.

9.2.2.3.1 Package ID Setting

The package ID can be set either from the NVM *Package ID* field in the NC-SI Configuration 1 NVM word ([Section 6.2.37.4](#)) or from an SDP pin. If set from SDP, the package ID is (0b, SDP value, 0b). The mode used is set by the *Read NCSI Package ID from SDP* field in the NC-SI Configuration 2 NVM word ([Section 6.2.39.5](#)). Note that when the package ID is set from the SDP pins, the used SDP should be set as an input in the relevant GLGEN_GPIO_CTL register. The SDP to use is defined in the *PackageID SDP* field in the NC-SI Configuration 2 NVM word ([Section 6.2.39.5](#)).

9.2.2.3.2 Channel ID Mapping

The mapping of the channels to physical ports is according to the NC-SI Channel to Port Mapping NVM word ([Section 6.2.39.14](#)) if the NC-SI Channel to Port Mapping Table *Valid* bit is set. If this bit is not set, the following algorithm should be used:

```
Channel_ID = 0
NC-SI_channel[3:0] = -1 // ports not associated to channels yet.
For func = 0 to 16 { // loop on all functions
    Port_ID = PFGEN_PORTNUM[func] // Port associated with function.
    If (PRTGEN_STATUS.PORT_VALID[Port_ID] && NC-SI_channel[Port_ID] == -1
    // Port is valid and port is not already associated to a channel
    NC-SI_channel[Port_ID] = Channel_ID; // assign channel
    Channel_ID++; // go to next channel
}
```



}

This algorithm maps channel numbers that match the order of the PCI function numbers.

Note: When a non standard mapping is used, ports should not be disabled via the disable straps.

9.3 Packet Filtering

Since both the host operating system and an MC use the X710-TM4 to send and receive Ethernet traffic, there needs to be a mechanism by which incoming Ethernet packets can be identified as those that should be sent to the MC rather than the host operating system.

A high level description of the host and MC filtering flow can be found in [Section 9.3.6](#).

There are two different types of filtering available. The first is filtering based upon the MAC address. With this filtering, the BMC has at least one dedicated MAC address and incoming Ethernet traffic with the matching MAC address(es) are passed to the MC. This is the simplest filtering mechanism to use and it enables the MC to receive all types traffic (including, but not limited to, IPMI, NFS, HTTP etc).

The other type available uses a highly configurable mechanism by which packets can be filtered using a wide range of parameters. Using this method, the MC can share a MAC address (and IP address, if desired) with the host operating system and receive only specific Ethernet traffic. This method is useful if the MC is only interested in specific traffic, such as IPMI packets.

9.3.1 Manageability Receive Filtering

This section describes the manageability receive packet filtering flow. Packet reception by the X710-TM4 can generate one of the following results:

- Discarded
- Sent to host memory
- Sent to the MC
- Sent to both the MC and host memory

The decisions regarding forwarding of packets to the host and to the MC are separate and are configured through two sets of registers. However, the MC might define some types of traffic as exclusive. This traffic is forwarded only to the MC, even if it passes the filtering process of the host. These types of traffic are defined using the PRT_MNG_MNGONLY register.

An example of packets that might be necessary to send exclusively to the MC might be specific TCP/UDP ports of a shared MAC address or a MAC address dedicated to the MC. If the MC configures the manageability filters to send these ports to the MC, it should configure the settings to not send them to the host; otherwise, these ports are received and handled by the host operating system.

The MC controls the types of packets that it receives by programming receive manageability filters. The following filters are accessible to the MC:



Table 9-5. Filters accessible to MC

Filters	Functionality	When Reset?
Filters Enable	General configuration of manageability filters	LAN_PWR_GOOD
Manageability Only	Enables routing of packets exclusively to manageability.	LAN_PWR_GOOD
Manageability Decision Filters [7:0]	Configuration of manageability decision filters	LAN_PWR_GOOD
MAC Address [3:0]	Four unicast MAC manageability addresses	LAN_PWR_GOOD
VLAN Filters [7:0]	Eight VLAN tag values	LAN_PWR_GOOD
UDP/TCP Port Filters [15:0]	16 destination port values	LAN_PWR_GOOD
Flexible 128 bytes TCO Filter	Length and values for one flex TCO filter	LAN_PWR_GOOD
IPv4 and IPv6 Address Filters [3:0]	IP address for manageability filtering	LAN_PWR_GOOD
Special filters modifier	Used to define some special filtering options like 24-bit filtering of IPv6 addresses and TCP/UDP selection of ASF ports	LAN_PWR_GOOD

All filtering capabilities are available on both the NC-SI and legacy SMBus interfaces. In NC-SI modes, part of the filters are programmed via standard NC-SI commands and part of the filter are programmed via the Intel OEM commands described in [Section 9.6.4](#). In legacy SMBus, the filtering is programmed either from NVM or via the commands described in [Section 9.5.10.1](#).

All filters are reset only on internal power on reset. Register filters that enable filters or functionality are also reset by firmware at internal firmware reset in NC-SI mode. These registers can be loaded from the NVM following a reset in SMBus mode. See [Section 6.2.37](#) for description of their location in the NVM map.

The high-level structure of manageability filtering is done using two or three steps.

1. The packet is routed by the switch as described in [Section 7.4.4.8](#). If the switch determines the packet should be routed to the manageability VSIs, the next steps are taken.
2. The packet is parsed and fields in the header are compared to programmed filters.
3. A set of decision filters are applied to the result of the first step.

The following sections describe steps 2 and 3 previously listed.

Some general rules apply:

- Fragmented packets are passed to manageability but not parsed beyond the IP header.
- Packets with L2 errors (CRC, alignment, etc.) are not forwarded to the MC.
- Packets longer than 2 KB are filtered out.
- All the filters refers to the outer header of the packet and not to any tunneled header.

The following sections describe the manageability filtering, followed by the final filtering rules.

The filtering rules are created by programming the decision filters as described in [Section 9.3.4](#).



9.3.2 L2 Filters

9.3.2.1 MAC and VLAN Filters

The manageability MAC filters allow a comparison of the destination MAC address to one of 4 filters defined in the *PRT_MNG_MMAH* and *PRT_MNG_MMAL* registers.

The VLAN filters allow a comparison of the 12 bit VLAN tag to one of 8 filters defined in the *PRT_MNG_MAVTV* registers. The VLAN tag compared is the innermost VLAN in the external L2 header.

9.3.2.2 EtherType Filters

Manageability L2 EtherType filters enable filtering of received packets based on the Layer 2 EtherType field. The L2 type field of incoming packets is compared against the EtherType filters programmed in the manageability EtherType filter (*PRT_MNG_METF*; up to 4 filters); the result is incorporated into decision filters.

Each manageability EtherType filter can be configured as pass (positive) or reject (negative) using a polarity bit. In order for the reverse polarity mode to be effective and block certain type of packets, the EtherType filter should be part of all the enabled decision filters.

An example for using L2 EtherType filters is to determine the destination of 802.1X control packets. The 802.1X protocol is executed at different times in either the management controller or by the host. L2 EtherType filters are used to route these packets to the proper agent.

In addition to the flexible EtherType filters, the X710-TM4 supports two fixed EtherType filters used to block NC-SI control traffic (0x88F8) and flow control traffic (0x8808) from reaching the manageability interface. The NC-SI EtherType is used for communication between the MC on the NC-SI link and the X710-TM4. Packets coming from the network are not expected to carry this EtherType and such packets are blocked to prevent attacks on the MC. Flow control packets should be consumed by the MAC and as such are not expected to be forwarded to the management interface.

Note: EtherType filters shouldn't configured with IPv4 or IPv6 Ethertype values.

9.3.3 L3/L4 Filtering

The manageability filtering stage combines checks done at previous stages with additional L3/L4 checks to make a decision about whether to route a packet to the MC. The following sections describe the manageability filtering done at layers L3/L4 and final filtering rules.

9.3.3.1 ARP Filtering

ARP filtering — The X710-TM4 supports filtering of ARP request packets (initiated externally) and ARP responses (to requests initiated by the MC).

In order to limit the reception of ARP packets to the ARP packets dedicated to this station (ARP target IP = MC IP), the ARP request/response filter can be bound to a specific IP address by setting both the ARP Request/Response and the IP AND bits in an MDEF filter. Note that the IP bit is also set if there is a match on the target IP (the TPA field in the ARP packet) of an ARP request or an ARP response.



Note: If the OR section of the MDEF is cleared and one of the IPv4 address are set, then ARP packets matching the IP address pass the filter. If these packets should be dropped, then an OR Ethertype filter with a a value of 0x0800 (IPv4) should be added.

See [Section 7.3.2.6](#) for the format of ARP packets.

9.3.3.2 Neighbor Discovery Filtering and MLD

The X710-TM4 supports filtering of the following ICMPv6 packets.

Neighbor discovery packets:

1. 0x86 (134d) - Router Advertisement.
2. 0x87 (135d) - Neighbor Solicitation.
3. 0x88 (136d) - Neighbor Advertisement.
4. 0x89 (137d) - Redirect.

MLD packets:

1. 0x82 (130d) - MLD Query
2. 0x83 (131d) - MLDv1 Report
3. 0x84 (132d) - MLD Done
4. 0x8F (143d) - MLDv2 Report

The neighbor discovery packets has dedicated enables for each type in the decision filters. For MLD, a single enable controls the forwarding of all the MLD packets. This means that either all the MLD packets types are selected for reception or none of them.

See [Section 7.3.2.5](#) for the format of ICMPv6 packets.

9.3.3.3 RMCP Filtering

The X710-TM4 supports filtering by fixed destination port numbers, port 0x26F and port 0x298. These ports are IANA reserved for RMCP.

UDP or TCP protocols can be included in the comparison using the *PRT_MNG_MSFM.PORT_26F/298_UDP/TCP* fields.

In SMBus mode, there are filters that can be enabled for these ports. When using NC-SI, they are not specifically available. However, the general filtering mechanism can be utilized to filter incoming RMCP traffic.

9.3.3.4 ICMP Filtering

The X710-TM4 supports filtering by ICMPv4. This filter matches if the IP *Protocol* field equals to 1b.

See [Section 7.3.2.4](#) for the format of ICMP packets.



9.3.3.5 Flexible Port Filtering

The X710-TM4 implements 16 flex source/destination port filters. The X710-TM4 directs packets whose L4 destination or source port matches to the MC. The MC must ensure that only valid entries are enabled in the decision filters.

For each flex port filter, filtering can be enabled for UDP, TCP or both. It can be enabled either on source or destination port.

9.3.3.6 IP Address Filtering

The X710-TM4 supports filtering by destination IP address using IPv4 and IPv6 address filters. These are dedicated to manageability. The X710-TM4 provides four IPv6 address filters and four IPv4 address filters.

For each IPv6 filter, the matching PRT_MNG_MSFM.IPV6_n_MASK bit defines if all the IP address should be compared to the PRT_MNG_MIPAF6 register or only the 24 LSBits should be compared to the 24 LSBits of the PRT_MNG_MIPAF6 register.

The IPv4 match also rises for ARP packets for which the target IP matches the IP address in the PRT_MNG_MIPAF4 register.

9.3.3.7 Checksum Filtering

To enable the XSUM filtering when using NC-SI, use the Enable Checksum Offloading command. See [Section 9.6.4.13](#).

Note: Checking the checksum of some complex IPv6 packets with routing extensions is not supported by X710-TM4. Hence such packets will be passed to the MC even if their checksum is wrong and checksum offload is enabled. The MC should check the checksum of such packets itself.

9.3.4 Flexible 128-byte Filter

The X710-TM4 provides one flex TCO filter. This filter looks for a pattern match within the first 128 bytes of the packet.

From the first 128 bytes, some of the fields are skipped for this comparison. These are field that are not exposed to the MC. The tag skipped is S-tag.

The flex filter programming should ignore the presence of this field.

Note: The flex filter comparison should be disabled in the MDEF registers while the flex filter is being updated.

Note: The flexible filter is not applied to transmit packets and a transmit packet is considered as if it didn't pass the filter.



9.3.4.1 Flexible Filter Structure

The filter is composed of the following fields:

1. Flexible filter length — This field indicates the number of bytes in the packet header that should be inspected. The field also indicates the minimal length of packets inspected by the filter. Packet below that length is not inspected. Valid values for this field are: $8*n$, where $n=1...16$.
2. Data — This is a set of up to 128 bytes comprised of values that header bytes of packets are tested against.
3. Mask — This is a set of bit masks - one bit per byte of data in the packet that indicate if is tested against its corresponding byte in the filter. Part of the bytes in the filter may not be used for the filtering, so the mask is used to indicate which of the bytes are used for the filter.

9.3.4.2 TCO Filter Programming

Programming each filter is done using the following commands (NC-SI or SMBus) in a sequential manner:

1. Filter Mask and Length — This command configures the following fields:
 - a. Mask — A set of 16 bytes containing the 128 bits of the mask. Bit 0 of the first byte corresponds to the first byte on the wire.
 - b. Length — A 1-byte field indicating the length.
2. Filter Data — The filter data is divided into groups of bytes as follows:

Group	Test Bytes
0x0	0-29
0x1	30-59
0x2	60-89
0x3	90-119
0x4	120-127

Each group of bytes need to be configured using a separate command, where the group number is given as a parameter. The command has the following parameters:

- a. Group number — A 1-byte field indicating the current group addressed.
- b. Data bytes — Up to 30 bytes of test-bytes for the current group.

9.3.4.2.1 Flexible TCO Filter Configuration in NVM (Global MNG Offset 0x05)

This section describes the NVM module used to store the flex filter initial data in SMBus mode.

This module is pointed to by global offset 0x08 of the manageability module header section.



9.3.4.2.1.1 Section Header – Offset 0x0

Bits	Name	Default	Description
15:0	Block Length	0xC	Section length in words (including CRC word and length word).

9.3.4.2.1.2 Flexible Filter Length and Control – Offset 0x01

Bits	Name	Default	Description
15:8	Flexible Filter Length (bytes)		
7:5	Reserved		Reserved.
4	Last Filter		
3	Apply Filter to LAN 3		
2	Apply Filter to LAN 2		
1	Apply Filter to LAN 1		
0	Apply Filter to LAN 0		

9.3.4.2.1.3 Flexible Filter Enable Mask – Offset 0x02 – 0x09A

Bits	Name	Default	Description
15:0	Flexible Filter Enable Mask		

9.3.4.2.1.4 Flexible Filter Data – Offset 0x0A – 0x49

Bits	Name	Default	Description
15:0	Flexible Filter Data		

Note: This section loads all of the flexible filters. The control + mask + filter data are repeatable as the number of filters. Section length in offset 0 is for all filters.

9.3.4.2.1.5 Section Footer – Offset Block Length

Bits	Name	Default	Description
15:8	CRC 8		CRC8 of the previous section.
7:0	Reserved		Reserved.



9.3.5 Configuring Manageability Filters

There are a number of pre-defined filters that are available for the MC to enable, such as ARPs and IPMI ports 0x298 and 0x26F. These are generally enabled by setting the appropriate bit within the PRT_MNG_MANC register using specific commands.

For more advanced filtering needs, the MC has the ability to configure a number of configurable filters. It is a two-step process to use these filters. They must first be configured and then enabled.

9.3.5.1 Manageability Decision Filters

Manageability Decision Filters (MDEF) are a set of eight filters, each with the same structure. The filtering rule for each decision filter is programmed by the MC and defines which of the L2, VLAN, Ethertype and L2/L3 filters participate in decision making. Any packet that passes at least one rule is directed to manageability and possibly to the host.

The inputs to each decision filter are:

- Packet passed a valid management L2 exact address filter.
- Packet is a broadcast packet.
- Packet has a VLAN header and it passed a valid manageability VLAN filter.
- Packet matched one of the valid IPv4 or IPv6 manageability address filters.
- Packet is a multicast packet.
- Packet passed ARP filtering (request or response).
- Packet passed neighbor solicitation filtering.
- Packet passed MLD filtering.
- Packet passed 0x298/0x26F port filter.
- Packet passed a valid flex port filter.
- Packet passed a valid flex TCO filter.
- Packet is an ICMPv4 packet.
- Packet passed or failed an L2 EtherType filter.
- Packet passed or failed Flow Control or NC-SI L2 EtherType Discard filter.

The structure of each decision filter is shown in [Figure 9-3](#). A boxed number indicates that the input is conditioned by a mask bit defined in the MDEF register and MDEF_EXT register for this rule. Decision filter rules are as follows:

- At least one bit must be set in one of the two registers. If all bits are cleared (MDEF/MDEF_EXT = 0x0000), then the decision filter is disabled and ignored.
- All enabled AND filters must match for the decision filter to match. An AND filter not enabled in the MDEF/MDEF_EXT registers is ignored. If an AND filter is preceded by a OR filter, then at least one of the enabled OR inputs must match for the filter to pass.
- If no OR filter is enabled in the register, the OR filters are ignored in the decision (the filter might still match).
- If one or more OR filters are enabled in the register, then at least one of the enabled OR filters must match for the decision filter to match.

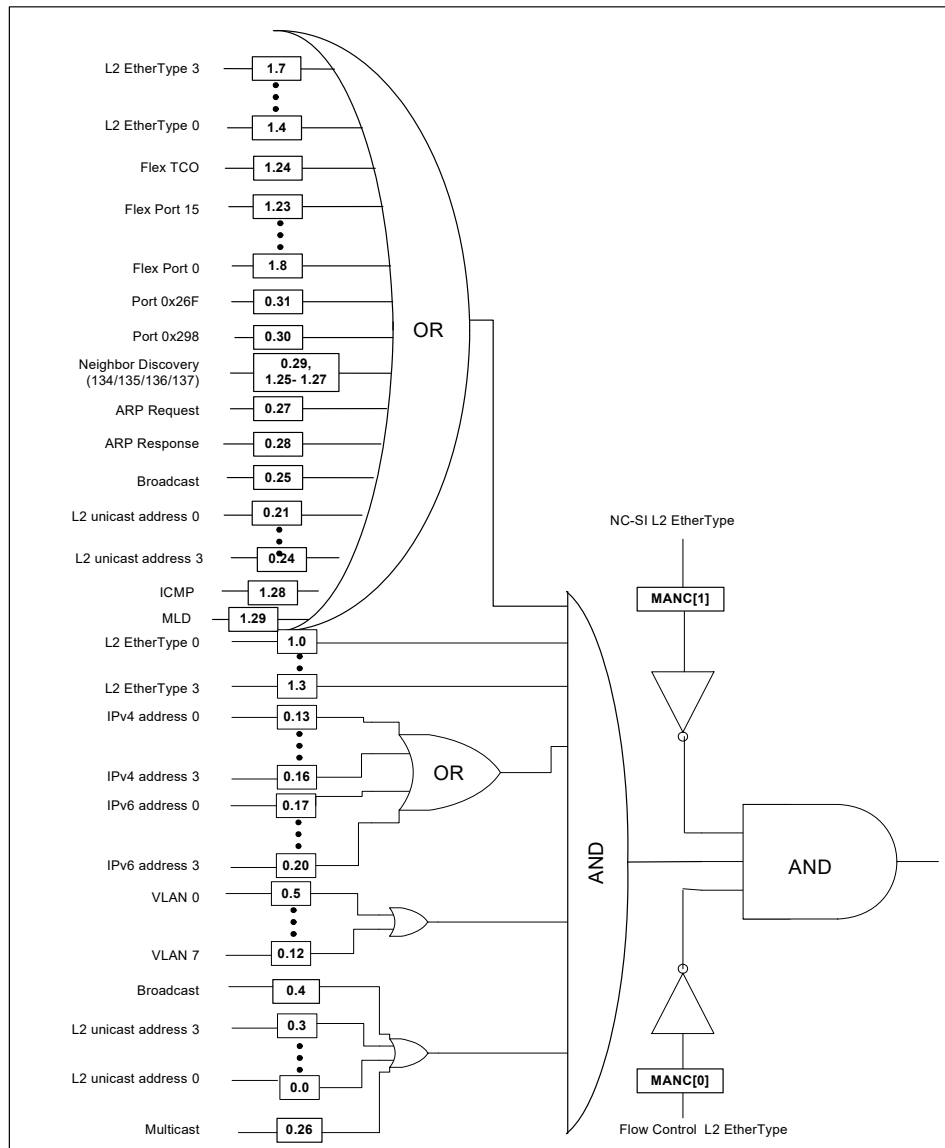


Figure 9-3. Manageability Decision Filters

A decision filter (for any of the eight filters) defines which of the previously described inputs are enabled as part of a filtering rule. The MC programs two 32-bit registers per rule (MDEF[7:0] and MDEF_EXT[7:0]) with the settings as described in [Section 10.2.1.21.4](#) and [Section 10.2.1.21.3](#). A set bit enables its corresponding filter to participate in the filtering decision.

In addition to the controls previously described, the *PRT_MNG_MDEF_EXT.apply_to_host_traffic* and *PRT_MNG_MDEF_EXT.apply_to_network_traffic* bits define which traffic is compared to this filter. At least one of these bits must be set for the filter to be valid.



If the *PRT_MNG_MDEF_EXT.apply_to_host_traffic* bit is set, the traffic from the host is a candidate for this filter. If the *PRT_MNG_MDEF_EXT.apply_to_network_traffic* bit is set, the traffic from the network is a candidate for this filter. If both bits are set, this filter is applied to all traffic.

9.3.5.2 Exclusive Traffic

The decisions regarding forwarding of packets to the host for LAN traffic or to the LAN for host traffic are independent from the management decision filters. However, the MC might define some types of traffic as exclusive. The behavior for such traffic is defined by the using the bits corresponding to the decision filter in the *PRT_MNG_MNGONLY* register (one bit per each of the eight decision rules) and the *PRT_MNG_MDEF_EXT.apply_to_host_traffic* and *PRT_MNG_MDEF_EXT.apply_to_network_traffic* bits. Table 9-6 lists the behavior in each case. If one or more filters match the traffic and at least one of the filters is set as exclusive, the traffic is treated as exclusive.

Table 9-6. Exclusive Traffic Behavior

traffic Source	Filter Match		Filter Doesn't Match
	PRT_MNG_MNGONLY = 0	PRT_MNG_MNGONLY = 1	N/A
From network	Traffic is forwarded to manageability. Traffic is forwarded to the host according to host filtering.	Traffic is forwarded only to manageability.	Traffic is forwarded to the host according to host filtering.
From host	Traffic is forwarded to manageability and to the LAN.	Traffic is forwarded only to manageability.	Traffic is forwarded to the LAN.

Any traffic matching any of the configurable filters (see Section 9.3.5.1) can be used as filters to pass traffic to the host.

Table 9-7. PRT_MNG_MNGONLY Register Description and Usage

Bits	Description	Default
0	Decision Filter 0	Determines if packets that have passed decision filter 0 are sent exclusively to the manageability path.
1	Decision Filter 1	Determines if packets that have passed decision filter 1 are sent exclusively to the manageability path.
2	Decision Filter 2	Determines if packets that have passed decision filter 2 are sent exclusively to the manageability path.
3	Decision Filter 3	Determines if packets that have passed decision filter 3 are sent exclusively to the manageability path.
4	Decision Filter 4	Determines if packets that have passed decision filter 4 are sent exclusively to the manageability path.
5	Unicast and Mixed	NC-SI mode: Determines if unicast and mixed packets are sent exclusively to the manageability path. SMBus mode: Determines if packets that have passed decision filter 5 are sent exclusively to the manageability path.
6	Global Multicast	NC-SI mode: Determines if multicast packets are sent exclusively to the manageability path. SMBus mode: Determines if packets that have passed decision filter 6 are sent exclusively to the manageability path.
7	Broadcast	NC-SI mode: Determines if broadcast packets are sent exclusively to the manageability path. SMBus mode: Determines if ARP packets are sent exclusively to the manageability path.
31:8	Reserved	Reserved.

When using the SMBus interface, the MC enables these filters by issuing the Update Management Receive Filter Parameters command with the parameter of 0x0F.



The *PRT_MNG_MNGONLY* is configurable when using NC-SI using the Set Intel Filters — Manageability Only Command (see [Section 9.6.4.5.3](#)).

All manageability filters are controlled by the MC only and not by the LAN software device driver.

9.3.5.3 Global Controls

On top of the *PRT_MNG_MDEF* filters, the *PRT_MNG_MANC* registers contain some global controls applied to all the packets in order to be a candidate for manageability filtering:

- Receive enable bits:
 - The *RCV_TCO_EN* field controls the reception of manageability traffic. It should be set only if one of the following bits is also set.
 - The *EN_BMC2OS* bit controls the reception of manageability traffic from the host.
 - The *EN_BMC2NET* bit controls the reception of manageability traffic from the network.
- VLAN filtering— In order to support the NC-SI VLAN modes the following controls are provided:
 - The *FIXED_NET_TYPE* field controls if only VLAN tagged or VLAN untagged traffic is received. If this bit is cleared both types are received. If it is set, only the type described by the *NET_TYPE* field is accepted.
 - If set, the *NET_TYPE* field indicates that only VLAN tagged traffic is received, if cleared only packets without VLAN is accepted. This field is validated by the *FIXED_NET_TYPE* field.

Both fields relates to the inner VLAN.

Table 9-8 lists the relationship between the previously mentioned bits and the forwarding decisions:

Table 9-8. PRT_MNG_MANC Bits Impact

CASE\ PRT_MNG_MANC Bits	RCV_TCO_EN=0b	FIXED_NET_TYPE= 1b and NET_TYPE!= What's in the Packet	EN_BMC2OS=0b (Assume EN_BMC2NET = 1b)	EN_BMC2NET=0b (Assume EN_BMC2HOST = 1b)
Packet sent from host and hits MDEF filters (host-to-MC traffic).	Packet is not sent to the MC.	Packet is not sent to the MC.	Packet is not sent to the MC.	Packet is sent to the MC.
Packet sent from host and matches one of the EMP VSI (host-to-EMP traffic).	Packet is sent to the EMP.	Packet is sent to the EMP.	Packet is sent to the EMP.	Packet is sent to the EMP.
Packet received from LAN and hits MDEF filters (LAN-to-MC traffic).	Packet is not sent to the MC.	Packet is not sent to the MC.	Packet is sent to the MC.	Packet is not sent to the MC.
Packet received from LAN and matches on of the EMP VSI (LAN-to-EMP traffic).	Packet is sent to the EMP.	Packet is sent to the EMP.	Packet is sent to the EMP.	Packet is sent to the EMP.
Packet sent from EMP and matches one of the host VSIs (EMP-to-host traffic)	Packet is sent to a host (and optionally to LAN).	Packet is sent to the host (and optionally to LAN).	Packet is sent to the host (and optionally to LAN).	Packet is sent to the host (and optionally to LAN).



Table 9-8. PRT_MNG_MANC Bits Impact

CASE\ PRT_MNG_MANC Bits	RCV_TCO_EN=0b	FIXED_NET_TYPE= 1b and NET_TYPE!= What's in the Packet	EN_BMC2OS=0b (Assume EN_BMC2NET = 1b)	EN_BMC2NET=0b (Assume EN_BMC2HOST = 1b)
Packet sent from the EMP and does not match one of the host VSIs (EMP-to-LAN traffic).	Packet is sent to the LAN.	Packet is sent to the LAN.	Packet is sent to the LAN.	Packet is sent to the LAN.
Packet sent from the MC and matches one of the host VSIs (MC-to-host traffic).	Packet is sent to the host (and optionally to LAN).	Packet is sent to the host (and optionally to LAN).	Packet is sent to the LAN.	Packet is sent to the host (and optionally to LAN).
Packet sent from the MC and does not match one of the host VSIs (MC-to-LAN traffic).	Packet is sent to the LAN.	Packet is sent to the LAN.	Packet is sent to the LAN.	Packet is not sent to the LAN (optionally sent to host).

9.3.6 Filtering Programming Interfaces

The X710-TM4 provides multiple options to program the forwarding filters, depending on the interface used and the level of flexibility needed. The [Table 9-9](#) lists the different options and points to the description of the relevant commands.

Table 9-9. Filtering Programming Interfaces

Interface	Flexible/Abstract	Description
NC-SI (over RMI or over MCTP)	Abstract (dedicated MAC address)	The regular NC-SI commands can be used to enable forwarding based on a dedicated MAC address. The list of supported commands can be found in Section 9.6.2.1 . When using these commands, one of the two other modes can be used to add finer grain filtering.
	Abstract (Shared MAC and IP)	The Intel OEM commands described in Section 9.3.6.1 and in Section 9.6.4.14 can be used to define which part of the shared MAC or shared IP traffic should be forwarded. When using these commands, the flexible filtering interface should not be used. This mode is activated using the Set Shared mode command (Section 9.6.4.14.12.1)
	Flexible	This interface described in most of the sub-sections of Section 9.6.4 . It uses the packet reduction commands to reduce the forwarding scope of the filters set by the regular NC-SI commands and the packet addition commands to add new packet types to the forwarding rules.
SMBus	Abstract	The Set Common filter command can be used to set the most common filters. When using this commands the flexible filtering interface should not be used. When sending this command, all previous filtering requests are cleared.
	Flexible	The Update MNG RCV Filter Parameters (Section 9.5.10.2.2) can be used to define the exact filtering rules to be applied.

9.3.6.1 Shared MAC and Shared IP Support

The X710-TM4 operates in systems where the same MAC and IP are shared between a platform's host operating system and its MC. In order to support such systems, the X710-TM4 supports additional shared MAC filtering options on top of what was supported in previous products. This section describes these options and the NC-SI commands used to program them.



Note: All filtering capabilities are exposed via the regular NC-SI packet reduction and packet addition commands and via the SMBus Set Filtering command. The interface described in this section is a more abstract NC-SI interface.

9.3.6.1.1 Sharing an IP and MAC Address

NC-SI over MCTP is used in desktop and mobile platforms. These platforms are typically used in enterprise environments outside of a data center. IP subnets in these environments are commonly designed such that more than 50% of their available addresses are assigned.

Hence, assigning a second IP address to an MC would generally necessitate a subnet redesign. Instead, a single IP address is typically shared between the host operating system and an MC in these platforms.

Because it's possible to bind multiple IP addresses to a single MAC address, the X710-TM4 needs to know the IP address shared by an MC in order to deliver packets to it. An MC uses the Set IP Address command to communicate its IP address to the X710-TM4. The Set IP Address command is defined in [Section 9.6.4.14.1](#).

In order to notify the X710-TM4 that the MC intends to use a shared MAC, the Set Shared Mode command ([Section 9.6.4.14.12.1](#)) should be given before programming any filter using the regular NC-SI commands (Set MAC address or Set VLAN) or the Intel OEM commands ([Section 9.6.4.14](#)).

9.3.6.1.1.1 TCP/UDP Ports Owned by an MC

A small subset of the TCP and UDP ports might be dedicated to an MC. The remaining ports are assigned to the host operating system. Hence, port-based filtering and the commands to configure it is required. For example, port-based filtering would be used to route WS-management packets to an MC.

The X710-TM4 needs to know the ports owned by an MC in order to deliver packets to it. An MC uses the Set Port command to communicate its ports to the X710-TM4. The Set Port command is defined in [Section 9.6.4.14.3](#). The X710-TM4 supports 10 port filters.

The Set Binding command is used to define the combination of MAC, VLAN, IP and ports that should be met to forward packets to the MC (see [Section 9.6.4.14.10](#) for more details).

9.3.6.1.1.2 Sharing Network Infrastructure Packets

In addition to management traffic, an MC needs to monitor network infrastructure traffic along with the host. For each flow, it is possible to define if it should include host traffic only, both host and network or only network.



9.3.6.1.1.3 ARP Filters Enhancement

ARP request message filtering is controlled by the Enable Broadcast Filter command. However, as currently defined, this command causes either all or no ARP requests to go to an MC. For MCTP over SMBus, attempting to forward all ARP requests within a subnet to an MC can easily overwhelm the available bandwidth. Therefore, an option to have the X710-TM4 forward only ARP requests that contain a Target IP Address value that matches the IP address used by an MC. An amendment to the Enable Broadcast Filter command is defined in the sections that follow to address this requirement.

9.3.7 Possible Configurations

This section describes ways of using management filters. Actual usage might vary.

9.3.7.1 Dedicated MAC Packet Filtering

- Select one of the eight rules for dedicated MAC filtering.
- Load the host MAC address to one of the management MAC address filters and set the appropriate bit in field 3:0 of the MDEF register.
- Set other bits to qualify which packets are allowed to pass through. For example:
 - Set bit 5 in the MDEF register to qualify with the first manageability VLAN.
 - Set relevant bits 13 to 20 in the MDEF register to qualify with a match to one of the IP addresses.
 - Set any L3/L4 bits (bits 27 to 31 in the MDEF register and bits 16 to 23 in MDEF_EXT) to qualify with any of a set of L3/L4 filters.

9.3.7.2 Broadcast Packet Filtering

- Select one of the eight rules for broadcast filtering.
- Set bit 25 in the MDEF register of the decision rule to enforce broadcast filtering.
- Set other bits to qualify which broadcast packets are allowed to pass through. For example:
 - Set bit 5 in the MDEF register to qualify with the first manageability VLAN.
 - Set relevant bits 13 to 20 in the MDEF register to qualify with a match to one of the IP addresses.
 - Set any L3/L4 bits (bits 27 to 31 in the MDEF register and bits 16 to 23 in MDEF_EXT) to qualify with any of a set of L3/L4 filters.

9.3.7.3 VLAN Packet Filtering

- Select one of the eight rules for VLAN filtering.
- Set bit 5 to 12 in the MDEF register to qualify with the relevant manageability VLANs.
- Set other bits to qualify which VLAN packets are allowed to pass through. For example:
 - Set any L3/L4 bits (bits 27 to 31 in the MDEF register and bits 16 to 23 in MDEF_EXT) to qualify with any of a set of L3/L4 filters.



9.3.7.4 IPv6 Filtering

IPv6 filtering is done using the following IPv6-specific filters:

- IP unicast filtering — requires filtering for link local address and a global address. Filtering setup might depend on whether or not the MAC address is shared with the host or dedicated to manageability:
 - Dedicated MAC address (for example, dynamic address allocation with DHCP does not support multiple IP addresses for one MAC address). In this case, filtering can be done at L2 using two dedicated unicast MAC filters.
 - Shared MAC address (for example, static address allocation sharing addresses with host). In this case, filtering needs to be done at L3, requiring two IPv6 address filters, one per address.
- A neighbor discovery filter — The X710-TM4 supports IPv6 neighbor discovery protocol. Since the protocol relies on multicast packets, the X710-TM4 supports filtering of these packets. IPv6 multicast addresses are translated into corresponding Ethernet multicast addresses in the form of 33-33-xx-xx-xx-xx, where the last 32 bits of the address are taken from the last 32 bits of the IPv6 multicast address. As a result, two direct MAC filters can be used to filter IPv6 solicited-node multicast packets as well as IPv6 all node multicast packets.

9.3.7.5 Receiving Filtering with Shared IP

When using the legacy SMBus interface or the MCTP interface, it is possible to share the host MAC and IP address with an MC. This functionality is also available when using base NC-SI using Intel OEM commands.

When an MC shares the MAC and IP address with the host, receive filtering is based on identifying specific flows through port allocation. The following setting might be used when using the legacy SMBus interface:

- Select one of the eight rules.
- Set a manageability dedicated MAC filter to the host MAC address and set the matching bit (0-3) in the MDEF register.
- If VLAN is used for management, load one or more management VLAN filters and set the matching bit (5- 12) in the MDEF register.

ARP filter/neighbor discovery filter is enabled when an MC is responsible for handling the ARP protocol. Set bit 27 or bit 28 in the MDEF register for this functionality.

In NC-SI over MCTP, dedicated commands are used to enable shared IP filtering.

9.3.8 Determining Manageability MAC Address

If an MC needs to use a dedicated MAC address or configure the automatic ARP response mechanism (only available in SMBus mode), it might be beneficial for an MC to be able to determine the MAC address used by the host.

Both the NC-SI and SMBus interfaces provide an Intel OEM command to read the system MAC address.



A possible use for this is that the MAC address programmed at manufacturing time does not increment by one each time, but rather by two. In this way, an MC can read the system MAC address and add one to it and be guaranteed of a unique MAC address.

Note: Determining the IP address being used by the host is beyond the scope of this document.

9.4 OS-to-BMC Traffic

9.4.1 Overview

Traditionally, the communication between a host and a local MC is not handled through the network interface and requires a dedicated interface such as an IPMI KCS interface. The X710-TM4 enables the host and the local MC communication via the regular pass-through interface, and thus enable management of a local console using the same interface used to manage any MC in the network.

When this flow is used, the host sends packets to an MC through the network interface. The X710-TM4 examines these packets and it then decides if they should be forwarded to an MC. On the inverse path, when an MC sends a packet on the pass-through interface, the X710-TM4 checks if it should be forwarded to the network, the host, or both. [Figure 9-4](#) describes the flow for OS-to-BMC traffic for the NC-SI over RBT case. OS2BMC is also available when operating over MCTP. It is not available in legacy SMBus mode.

The OS-to-BMC flow can be enabled using the *OS2BMC Enable* field for the relevant port in the OS-to-BMC configuration structure of the NVM.

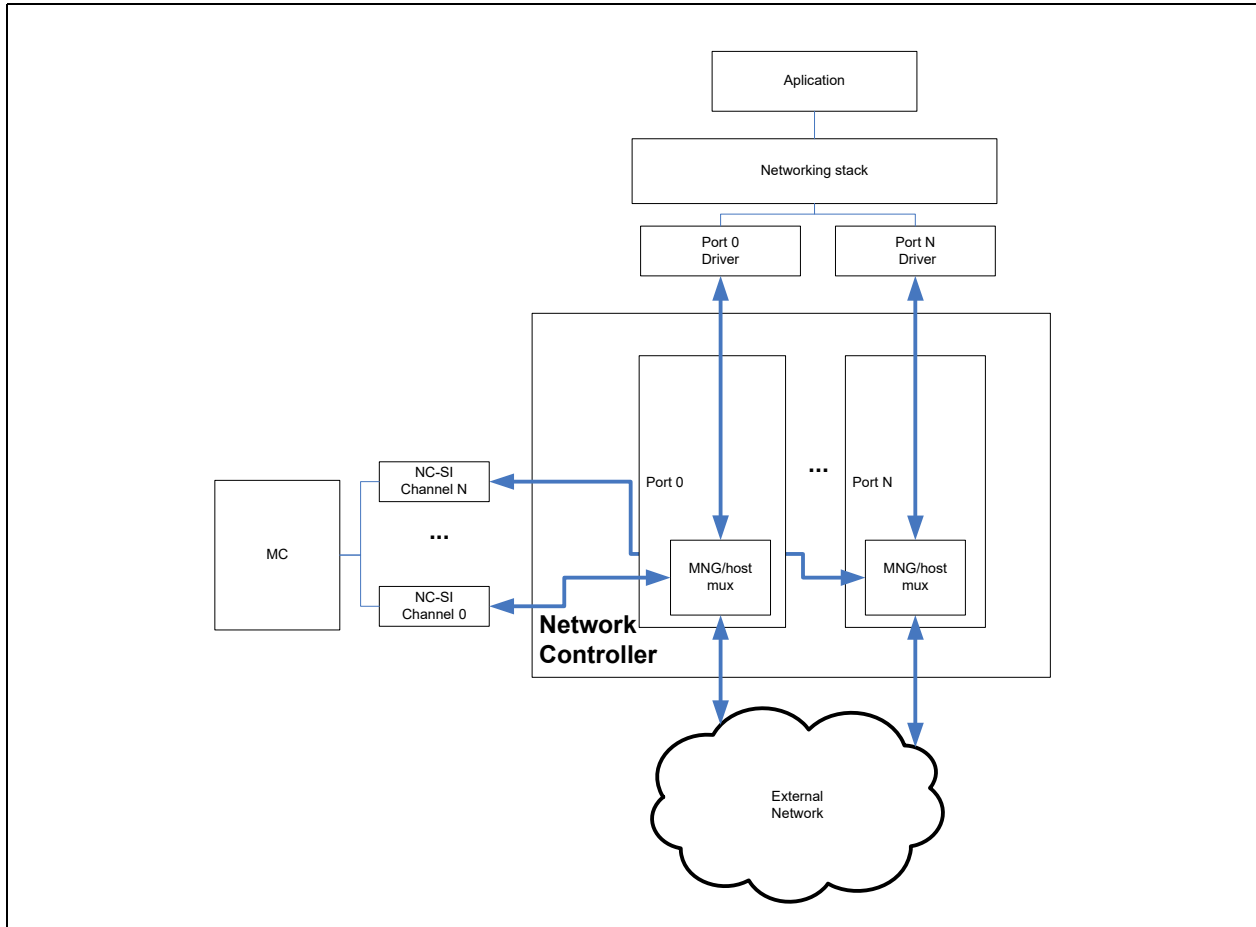


Figure 9-4. OS-to-BMC Flow

The OS-to-BMC flow is enabled only for ports enabled by the NC-SI Enable Channel command or via the *OS2BMC Enable* field for the relevant port in the OS-to-BMC configuration structure of the NVM.

OS-to-BMC traffic must comply with NC-SI specifications and is therefore limited to maximum sized frames of 1536 bytes (in both directions).

9.4.2 Filtering

9.4.2.1 OS-to-BMC Filtering

The flow used to filter packets from the MC to the host is described in [Section 7.4.4.8.2](#).

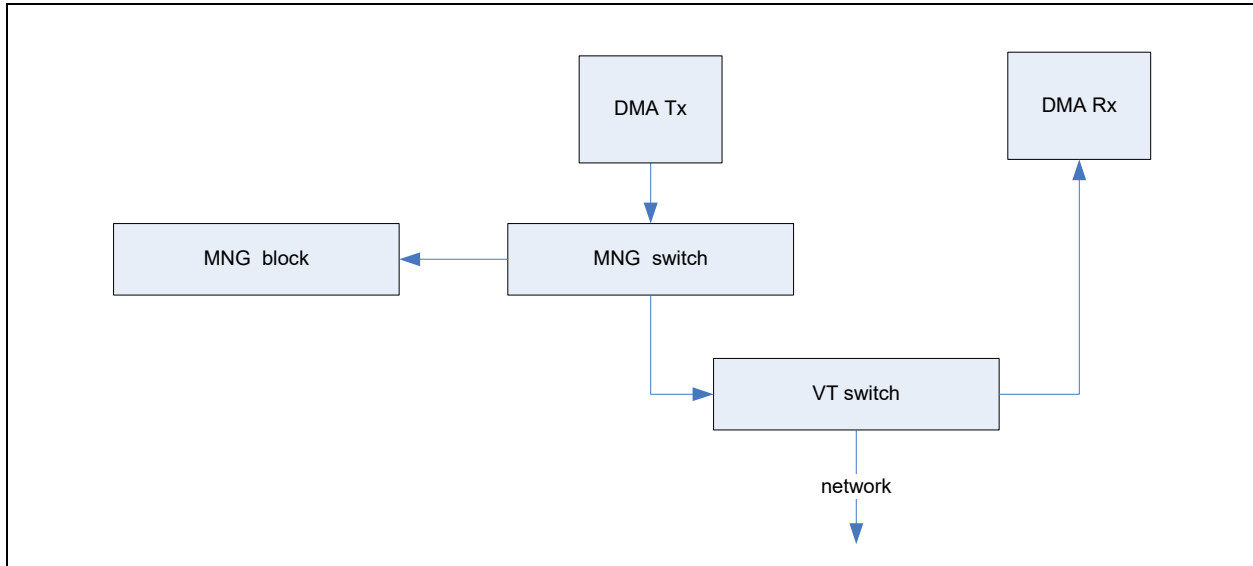


Figure 9-5. OS-to-BMC and VM-to-VM Filtering

9.4.2.2 BMC-to-OS Filtering

The flow used to filter packets from the BMC to the host is similar to the filtering of receive traffic (described in [Section 7.4.4.8.3](#)).

Note: Traffic sent from the MC does not cause a PME event, even if it matches one of the wake-up filters set by the port.

9.4.3 Blocking Network-to-BMC Flow

In some systems the MC might have its own private connection to the network and might use a the X710-TM4 port only for the OS-to-BMC traffic. In this case, the BMC-to-network flow should be blocked while enabling the OS-to-BMC and OS-to-network flows.

This can be done by clearing the *PRT_MNG_MANC.EN_BMC2NET* bit for the relevant port. The MC can control this functionality using the Enable Network-to-BMC flow and Disable Network-to-BMC flow NC-SI OEM commands. This can also be controlled using the *Network to BMC disable* field in the NVM OS2BMC configuration structure.

Note: The NC-SI channel should not be enabled for receive or transmit before at least one of the *PRT_MNG_MANC.EN_BMC2NET* or *PRT_MNG_MANC.EN_BMC2OS* fields is set, unless only used for AEN transmissions. In this case, the channel might be enabled for receive, but all receive filters should be cleared.



9.4.4 OS-to-BMC and Flow Control

The traffic between the host and manageability uses the same buffers as any loopback traffic. Thus, it flows through the transmit buffer and then through the receive buffer. If the transmit buffer is flow controlled, then the host-to-MC traffic is also stopped. If the receive buffer is full, the traffic is dropped or the transmit is stopped according to the flow control policy of this traffic class.

Packets received by manageability (either from the host or from the network) might be dropped if the manageability internal buffers are full.

9.4.5 Statistics

Packets sent from the operating system to the MC should be counted by all statistical counters as packets sent by the operating system. If they are sent to both the network and to the MC, then they are counted once.

Packets sent from the MC to the host are counted as packets received by the host. If they are sent to the host and to the network, then they are counted both as received packets and as packet transmitted to the network.

See [Section 7.11.5](#) for details of the statistics hierarchy.

9.4.6 OS-to-BMC Enablement

The X710-TM4 supports the unified network software model for OS-to-BMC traffic, where the OS-to-BMC traffic is shared with the regular traffic. In this model, there is no need for a special configuration of the operating networking stack or the BMC stack, but if the link is down, then the OS-to-BMC communication is stopped.

In order to enable OS-to-BMC either:

- Enable *OS2BMC* in the port traffic type field in the Traffic Type Parameters NVM word for the relevant port.
- Send an Enable Network-to-BMC command.

Note: When *OS2BMC* is enabled, the operating system must avoid sending packets longer than 1.5 KB to the MC. Such packets are dropped.

9.5 SMBus Pass Through Interface

SMBus is the system management bus defined by Intel. It is used in personal computers and servers for low-speed system management communications. This section describes how the SMBus interface operates in legacy PT mode.



9.5.1 General

The SMBus sideband interface includes standard SMBus commands used for assigning a slave address and gathering device information as well as Intel proprietary commands used specifically for the pass-through interface.

9.5.2 PT Capabilities

This section details manageability capabilities the X710-TM4 provides while in SMBus mode. PT traffic is carried by the sideband interface as described in [Section 9.1](#).

These services are not available in NC-SI mode.

When operating in SMBus mode, in addition to exposing a communication channel to the LAN for the MC, the X710-TM4 provides the following manageability services to the MC:

- ARP handling — The X710-TM4 can be programmed to auto-ARP replying for ARP request packets to reduce the traffic over the MC interconnect.
- Default configuration of filters by NVM — When working in SMBus mode, the default values of the manageability receive filters can be set according to the PT LAN and flex TCO NVM structures.
- Padding of short packets. Packets smaller than 60 bytes but larger than 16 bytes are padded to a legal Ethernet packet.
- CRC calculation — The X710-TM4 adds an Ethernet CRC on all sent packets.

9.5.3 Port-to-SMBus Mapping

The X710-TM4 is identified on the SMBus manageability link as four different devices via four different SMBus addresses on which each device is connected to a different LAN port. There is no logical connection between the four devices.

Only the ports enabled for manageability are exposed as SMBus devices.

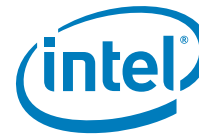
The fail-over between the LAN ports is done by the MC (by sending/receiving packets through different devices). The status report to the MC, ARP handling, DHCP, and other pass-through functionality are unique for each port and configured by the MC.

9.5.4 Automatic Ethernet ARP Operation

The X710-TM4 can offload the Ethernet Address Resolution Protocol (ARP) for the MC in order to reduce the bandwidth required on the SMBus link.

Automatic Ethernet ARP parameters are loaded from the NVM when the X710-TM4 is powered up or configured through the sideband management interface. The following parameters should be configured in order to enable ARP operation:

- ARP auto-reply enabled
- ARP IP address (to filter ARP packets)
- ARP MAC addresses (for ARP responses)



These are all configurable over the sideband interface using the advanced version of the Receive Enable command.

When an ARP request packet is received and ARP auto-reply is enabled, the X710-TM4 checks the targeted IP address (after the packet has passed L2 checks and ARP checks). If the targeted IP matches the IP configuration for the X710-TM4, it replies with an ARP response.

The X710-TM4 responds to ARP request targeted to the ARP IP address with the configured ARP MAC address. In case that there is no match, the X710-TM4 silently discards the packets. If the X710-TM4 is not configured to do an auto-ARP response, it can be configured to forward the ARP packets to the MC, which can respond to ARP requests.

When the external MC uses the same IP and MAC address of the operating system, the ARP operation should be coordinated with the host operating system.

Note: If sharing the MAC and IP with the host operating system is possible, the X710-TM4 provides the ability to read the system MAC address, enabling the MC to share the MAC address. However, there is no mechanism provided by the X710-TM4 to read the IP address. The host operating system (or an agent within) and the MC must coordinate the sharing of IP addresses.

9.5.5 SMBus Notification Methods

The X710-TM4 supports three methods of notifying the MC that it has information that needs to be read by the MC:

- SMBus alert — Refer to [Section 9.5.5.1](#).
- Asynchronous notify — Refer to [Section 9.5.5.2](#).
- Direct receive — refer to [Section 9.3.5.3](#).

The notification method used by the X710-TM4 can be configured from the SMBus using the Receive Enable command ([Section 9.5.10.1.3](#)). The default method is set by the NVM in the *Notification Method* field in LAN Receive Enable 1 ([Section 6.2.32.9](#)).

Note: The SMBus notification method used must be the same for all ports.

The following events cause the X710-TM4 to send a notification event to the MC:

- Receiving a LAN packet that is designated to the MC.
- Firmware was reset and requires re-initialization.
- Receiving a Request Status command from the MC initiates a status response.
- The X710-TM4 is configured to notify the MC upon status changes (by setting the EN_STA bit in the Receive Enable command) and one of the following events happen:
 - TCO command aborted
 - Link status changed
 - Power state change

There can be cases where the MC is hung and not responding to the SMBus notification. The X710-TM4 has a time-out value (defined in the NVM) to avoid hanging while waiting for the notification response. If the MC does not respond until the time out expires, the notification is de-asserted and all pending data is silently discarded.

Note that the SMBus notification time-out value can only be set in the NVM. The MC cannot modify this value.



9.5.5.1 SMBus Alert and Alert Response Method

The SMBus alert (SMBALRT_N) signal is an additional SMBus signal that acts as an asynchronous interrupt signal to an external SMBus master. The X710-TM4 asserts this signal each time it has a message that it needs the MC to read and if the chosen notification method is the SMBus alert method. Note that the SMBus alert method is an open-drain signal which means that other devices besides the X710-TM4 can be connected on the same alert pin. As a result, the MC needs a mechanism to distinguish between the alert sources.

The MC can respond to the alert by issuing an ARA cycle command to detect the alert source device. The X710-TM4 responds to the ARA cycle with its own SMBus slave address (if it was the SMBus alert source) and de-asserts the alert when the ARA cycle is completes. Following the ARA cycle, the MC issues a read command to retrieve the X710-TM4 message.

Some MCs do not implement the ARA cycle transaction. These MCs respond to an alert by issuing a Read command to the X710-TM4 (0xC0/0xD0 or 0xDE). The X710-TM4 always responds to a Read command, even if it is not the source of the notification. The default response is a status transaction. If the X710-TM4 is the source of the SMBus Alert, it replies the read transaction and then de-asserts the alert after the command byte of the read transaction.

Note: In SMBus alert mode, the SMBALRT_N pin is used for notification. Each port generate alerts on events that are independent of each other.

Note: If two ports have events to notify, the second alert is asserted only after the first event is handled. Hence, if an ARA cycle is not sent, a read transaction should be done to all the ports of the X710-TM4 in event of an ALERT_N assertion.

The ARA cycle is an SMBus receive byte transaction to SMBus Address 0001b - 100b. Note that the ARA transaction does not support PEC. The ARA transaction format is as follows:

1	7	1	1	8	1	1	1
S	Alert Response Address	Rd	A	Slave Device Address		A	P
	0001 100	1	0	Manageability Slave SMBus Address	0	1	

Note: If the MC does not react to the alert in the delay defined by the *SMBus Notification Timeout* NVM field (Section 6.2.39.3), the ALERT pin is de-asserted and the Rx packet indication in the status word is cleared (and packet is dropped).

9.5.5.2 Asynchronous Notify Method

When configured using the asynchronous notify method, the X710-TM4 acts as a SMBus master and notifies the MC of one of the events listed in Section 9.5.5 by issuing a modified form of the write word transaction. The asynchronous notify transaction SMBus address and data payload is configured using the Receive Enable command (Section 9.5.10.1.3) or using the NVM defaults. Note that the asynchronous notify is not protected by a PEC byte.



1	7	1	1	7	1	1	8	1	8	1	1
S	Target Address	Wr	A	Sending Device Address		A	Data Byte Low	A	Data Byte High	A	P
	MC Slave Address	0	0	MNG Slave SMBus Address	0	0	Interface	0	Alert Value	0	

The target address and data byte low/high are taken from the Receive Enable command or NVM configuration (Section 6.2.37.9 and Section 6.2.37.10).

If the MC does not read the status in the delay defined by the *SMBus Notification Timeout* NVM field (Section 6.2.39.3), the Rx packet indication in the status word is cleared (and packet is dropped).

9.5.5.3 Direct Receive Method

If configured, the X710-TM4 has the capability to send a message it needs to transfer to the external MC as a master over the SMBus instead of alerting the MC and waiting for it to read the message.

The message format follows. Note that the command that is used is the same command that is used by the external MC in the Block Read command. The opcode that the X710-TM4 puts in the data is also the same as it put in the Block Read command of the same functionality. The rules for the *F* and *L* flags (bits) are also the same as in the Block Read command.

1	7	1	1	1	1	6	1	
S	Target Address	Wr	A	F	L	Command	A	...
	BMC Slave Address	0	0	First Flag	Last Flag	Receive TCO Command 01 0000b	0	

8	1	8	1		1	8	1	1
Byte Count	A	Data Byte 1	A	...	A	Data Byte N	A	P
N	0		0		0		0	

9.5.6 Receive PT Flow

The X710-TM4 is used as a channel for receiving packets from the network link and passing them to the external MC. The MC configures the X710-TM4 to pass these specific packets to the MC. Once a full packet is received from the link and identified as a manageability packet that should be transferred to the MC, the X710-TM4 starts the receive TCO flow to the MC.

The X710-TM4 uses the SMBus notification method to notify the MC that it has data to deliver. Since the packet size might be larger than the maximum SMBus fragment size, the packet is divided into fragments, where the X710-TM4 uses the maximum fragment size allowed in each fragment (configured via the NVM). The last fragment of the packet transfer is always the status of the packet. As a result, the packet is transferred in at least two fragments. The data of the packet is transferred as part of the receive TCO LAN packet transaction.



When SMBus alert is selected as the MC notification method, the X710-TM4 notifies the MC on each fragment of a multi-fragment packet. When asynchronous notify is selected as the MC notification method, the X710-TM4 notifies the MC only on the first fragment of a received packet. It is the MC's responsibility to read the full packet including all the fragments.

Any timeout on the SMBus notification results in discarding the entire packet. Any NACK by the MC causes the fragment to be re-transmitted to the MC on the next Receive Packet command.

The maximum size of the received packet is limited by the X710-TM4 to 1536 bytes. Packets larger than 1536 bytes are silently discarded. Any packet smaller than 1536 bytes is processed.

9.5.7 Transmit PT Flow

The X710-TM4 is used as the channel for transmitting packets from the external MC to the network link. The network packet is transferred from the MC over the SMBus and then, when fully received by the X710-TM4, is transmitted over the network link.

Each SMBus address is connected to a different LAN port. When a packet is received during a SMBus transaction using SMBus address #0, it is transmitted to the network using LAN port #0; it is transmitted through LAN port #1 if received on SMBus address #1, etc.

The X710-TM4 supports packets up to an Ethernet packet length of 1536 bytes. Since SMBus transactions can only be up to 240 bytes in length, packets might need to be transferred over the SMBus in more than one fragment. This is achieved using the F and L bits in the command number of the transmit TCO packet Block Write command. When the F bit is set, it is the first fragment of the packet. When the L bit is set, it is the last fragment of the packet. When both bits are set, the entire packet is in one fragment. The packet is sent over the network link only after all its fragments are received correctly over the SMBus. The maximum SMBus fragment size is defined within the NVM and cannot be changed by the BMC.

The minimum packet length defined by the 802.3 spec is 64 bytes. The X710-TM4 pads packets that are less than 64 bytes to meet the specification requirements (there is no need for the external MC to pad packets less than 64 bytes). If the packet sent by the MC is larger than 1536 bytes, the X710-TM4 silently discards the packet. The minimal packet size that the X710-TM4 can handle is 14 bytes.

The X710-TM4 calculates the L2 CRC on the transmitted packet and adds its four bytes at the end of the packet. Any other packet field (such as XSUM or VLAN) must be calculated and inserted by the MC (the X710-TM4 does not change any field in the transmitted packet, other than adding padding and CRC bytes).

If the network link is down when the X710-TM4 has received the last fragment of the packet from the MC, it silently discards the packet. Note that any link down event during the transfer of any packet over the SMBus does not stop the operation since the X710-TM4 waits for the last fragment to end to see whether the network link is up again.

9.5.7.1 Transmit Errors in Sequence Handling

Once a packet is transferred over the SMBus from the MC to the X710-TM4, the F and L flags should follow specific rules. The F flag defines the first fragment of the packet; the L flag that the transaction contains the last fragment of the packet. [Table 9-8](#) lists the different flag options in transmit packet transactions.

**Table 9-10. Flag Options During Transmit Packet Transactions**

Previous	Current	Action/Notes
Last	First	Accept both.
Last	Not First	Error for the current transaction. Current transaction is discarded and an abort status is asserted.
Not Last	First	Error in previous transaction. Previous transaction (until previous First) is discarded. Current packet is processed. No abort status is asserted.
Not Last	Not First	Process the current transaction.

Note: Since every other Block Write command in TCO protocol has both *F* and *L* flags set, they cause flushing any pending transmit fragments that were previously received. When running the TCO transmit flow, no other Block Write transactions are allowed in between the fragments.

9.5.7.2 TCO Command Aborted Flow

The X710-TM4 indicates to the MC an error or an abort condition by setting the *TCO Abort* bit in the general status. The X710-TM4 might also be configured to send a notification to the MC (see [Section 9.5.10.1.6](#)).

Following is a list of possible error and abort conditions:

- Any error in the SMBus protocol (NACK, SMBus timeouts, etc.).
- If the MC does not respond until the notification timeout (programmed in the NVM) expires.
- Any error in compatibility between required protocols to specific functionality (for example, Rx Enable command with a byte count not equal to 1/14, as defined in the command specification).
- If the X710-TM4 does not have space to store the transmitted packet from the MC (in its internal buffer space) before sending it to the link, the packet is discarded and the external MC is notified via the *Abort* bit.
- Error in the *F/L* bit sequence during multi-fragment transactions.
- An internal reset to the X710-TM4's firmware.

9.5.8 SMBus Link State Control

While in SMBus mode, the default setting of the link is defined by the *EMP_LINK_ON* bit in Common Firmware Parameters 2 NVM word.

When a channel is enabled through NVM setting or through the *RCV_EN* option of the Receive Enable command, the link is established (if not already required for other purposes).

If the channel is disabled by clearing of the *RCV_EN* option, then the link might move back to the default defined by the *EMP_LINK_ON* if not needed for other purposes.

Note: Before transitioning to D3 it is the responsibility of the software device driver to request the PHY to be active for wake-up activities.



9.5.9 SMBus ARP Transactions

All SMBus ARP transactions include the PEC byte.

9.5.9.1 Prepare to ARP

This command clears the *Address Resolved* flag (set to false). It does not affect the status or validity of the dynamic SMBus address and is used to inform all devices that the ARP master is starting the ARP process:

1	7	1	1	8	1	8	1	1
S	Slave Address	Wr	A	Command	A	PEC	A	P
	1100 001	0	0	0000 0001	0	[Data Dependent Value]	0	

9.5.9.2 Reset Device (General)

This command clears the *Address Resolved* flag (set to false). It does not affect the status or validity of the dynamic SMBus address.

1	7	1	1	8	1	8	1	1
S	Slave Address	Wr	A	Command	A	PEC	A	P
	1100 001	0	0	0000 0010	0	[Data Dependent Value]	0	

9.5.9.3 Reset Device (Directed)

The Command field is NACKed if bits 7:1 do not match the current SMBus address. This command clears the *Address Resolved* flag (set to false) and does not affect the status or validity of the dynamic SMBus address.

1	7	1	1	8	1	8	1	1
S	Slave Address	Wr	A	Command	A	PEC	A	P
	1100 001	0	0	Targeted Slave Address 0	0	[Data Dependent Value]	0	

9.5.9.4 Assign Address

This command assigns SMBus address. The address and command bytes are always acknowledged.



The transaction is aborted (NACKed) immediately if any of the UDID bytes is different from X710-TM4 UDID bytes. If successful, the manageability system internally updates the SMBus address. This command also sets the *Address Resolved* flag (set to true).

1	7	1	1	8	1	8	1	
S	Slave Address	W	A	Command	A	Byte Count	A	...
	1100 001	0	0	0000 0100	0	0001 0001	0	

8	1	8	1	8	1	8	1	
Data 1	A	Data 2	A	Data 3	A	Data 4	A	...
UDID Byte 15 (MSB)	0	UDID Byte 14	0	UDID Byte 13	0	UDID Byte 12	0	

8	1	8	1	8	1	8	1	
Data 5	A	Data 6	A	Data 7	A	Data 8	A	...
UDID Byte 11	0	UDID Byte 10	0	UDID Byte 9	0	UDID Byte 8	0	

8	1	8	1	8	1	
Data 9	A	Data 10	A	Data 11	A	...
UDID Byte 7	0	UDID Byte 6	0	UDID Byte 5	0	

8	1	8	1	8	1	8	1	
Data 12	A	Data 13	A	Data 14	A	Data 15	A	...
UDID Byte 4	0	UDID Byte 3	0	UDID Byte 2	0	UDID Byte 1	0	

8	1	8	1	8	1	1
Data 16	A	Data 17	A	PEC	A	P
UDID Byte 0 (LSB)	0	Assigned Address	0	[Data Dependent Value]	0	



9.5.9.5 Get UDID (General and Directed)

The general get UDID SMBus transaction supports a constant command value of 0x03 and, if directed, supports a Dynamic command value equal to the dynamic SMBus address.

If the SMBus address has been resolved (*Address Resolved* flag set to true), the manageability system does not acknowledge (NACK) this transaction. If it's a General command, the manageability system always acknowledges (ACKs) as a directed transaction.

This command does not affect the status or validity of the dynamic SMBus address or the *Address Resolved* flag.

S	Slave Address	Wr	A	Command	A	S	...
	1100 001	0	0	See Below	0		

7	1	1	8	1	...
Slave Address	Rd	A	Byte Count	A	...
1100 001	1	0	0001 0001	0	

8	1	8	1	8	1	8	1	...
Data 1	A	Data 2	A	Data 3	A	Data 4	A	...
UDID Byte 15 (MSB)	0	UDID Byte 14	0	UDID Byte 13	0	UDID Byte 12	0	

8	1	8	1	8	1	8	1	...
Data 5	A	Data 6	A	Data 7	A	Data 8	A	...
UDID Byte 11	0	UDID Byte 10	0	UDID Byte 9	0	UDID Byte 8	0	

8	1	8	1	8	1	...
Data 9	A	Data 10	A	Data 11	A	...
UDID Byte 7	0	UDID Byte 6	0	UDID Byte 5	0	

8	1	8	1	8	1	8	1	...
Data 12	A	Data 13	A	Data 14	A	Data 15	A	...
UDID Byte 4	0	UDID Byte 3	0	UDID Byte 2	0	UDID Byte 1	0	



8	1	8	1	8	1	1
Data 16	A	Data 17	A	PEC	~Ä	P
UDID Byte 0 (LSB)	0	Device Slave Address	0	[Data Dependent Value]	1	

The Get UDID command depends on whether or not this is a Directed or General command.

The General Get UDID SMBus transaction supports a constant command value of 0x03.

The Directed Get UDID SMBus transaction supports a Dynamic command value equal to the dynamic SMBus address with the LSB bit set.

Note: Bit 0 (LSB) of Data byte 17 is always 1b.

9.5.10 SMBus PT Transactions

This section details commands (both read and write) that the X710-TM4 SMBus interface supports for PT.

9.5.10.1 Write SMBus Transactions

This section details the commands that the MC can send to the X710-TM4 over the SMBus interface. The SMBus write transactions table lists the different SMBus write transactions supported by the X710-TM4.

TCO Command	Transaction	Command	Fragmentation	Section
Transmit Packet	Block Write	First: 0x84 Middle: 0x04 Last: 0x44	Multiple	9.5.10.1.1
Transmit Packet	Block Write	Single: 0xC4	Single	9.5.10.1.1
Request Status	Block Write	Single: 0xDD	Single	9.5.10.1.2
Receive Enable	Block Write	Single: 0xCA	Single	9.5.10.1.3
Force TCO	Block Write	Single: 0xCF	Single	9.5.10.2
Management Control	Block Write	Single: 0xC1	Single	9.5.10.2.1
Update MNG RCV Filter Parameters	Block Write	Single: 0xCC	Single	9.5.10.2.2
Set Common Filters	Block Write	Single: 0xC2	Single	9.5.10.2.3
Clear All Filters	Block Write	Single: 0xC3	Single	9.5.10.2.4

9.5.10.1.1 Transmit Packet Command

The Transmit Packet command behavior is detailed in [Section 9.5.7](#). The Transmit Packet fragments have the following format.



The payload length is limited to the maximum payload length set in the NVM. If the overall packet length is bigger than 1536 bytes, the packet is silently discarded.

Function	Command	Byte Count	Data 1	...	Data N
Transmit first fragment	0x84	N	Packet data MSB	...	Packet data LSB
Transmit middle fragment	0x04				
Transmit last fragment	0x44				
Transmit single fragment	0xC4				

9.5.10.1.2 Request Status Command

An external MC can initiate a request to read X710-TM4 manageability status by sending a Request Status command. When received, the X710-TM4 initiates a notification to an external MC when status is ready. After this, the external controller is able to read the status, by issuing a Read Status command (see [Section 9.5.10.3.2](#)).



The format is as follows:

Function	Command	Byte Count	Data 1
Request Status	0xDD	1	0

9.5.10.1.3 Receive Enable Command

The Receive Enable command is a single fragment command used to configure the X710-TM4. This command has two formats: short, 1-byte legacy format (providing backward compatibility with previous components) and long, 14-byte advanced format (allowing greater configuration capabilities). The Receive Enable command format is as follows:

Function	CMD	Byte Count	Data 1	Data 2	...	Data 7	Data 8	...	Data 11	Data 12	Data 13	Data 14
Legacy Receive Enable	0xCA	1	Receive Control Byte	-	...	-	-	...	-	-	-	-
Advanced Receive Enable		14 (0x0E)		MAC Addr MSB		MAC Addr LSB	IP Addr MSB		IP Addr LSB	BMC SMBus Addr	I/F Data Byte	Alert Value Byte

Field	Bit(s)	Description
RCV_EN	0	Receive TCO Enable. 0b = Disable receive TCO packets. 1b = Enable Receive TCO packets. Setting this bit enables all manageability receive filtering operations. Enabling specific filters is done via the NVM or through special configuration commands. Note: When the RCV_EN bit is cleared, all receive TCO functionality is disabled, not just the packets that are directed to the MC (also auto ARP packets).
RCV_ALL	1	Receive All Enable. 0b = Disable receiving all packets. 1b = Enable receiving all packets. Forwards all packets received over the wire that passed L2 filtering to the external MC. This flag has no effect if bit 0 (Enable TCO packets) is disabled.
EN_STA	2	Enable Status Reporting. 0b = Disable status reporting. 1b = Enable status reporting.



Field	Bit(s)	Description
EN_ARP_RES	3	<p>Enable ARP Response.</p> <p>0b = Disable the X710-TM4 ARP response.</p> <p>The X710-TM4 treats ARP packets as any other packet, for example, packet is forwarded to the MC if it passed other (non-ARP) filtering.</p> <p>1b = Enable the X710-TM4 ARP response.</p> <p>The X710-TM4 automatically responds to all received ARP requests that match the IP address programmed by the MC. The MC IP address is provided as part of the Receive Enable message (bytes 8:11). If a short version of the command is used, the X710-TM4 uses IP address configured in the most recent long version of the command in which the EN_ARP_RES bit was set. If no such previous long command exists, then the X710-TM4 uses the IP address configured in the NVM as ARP Response IPv4 address in the PT LAN configuration structure.</p> <p>If the <i>CBDM</i> bit is set, the X710-TM4 uses the BMC dedicated MAC address in ARP response packets. If the <i>CBDM</i> bit is not set, the MC uses the host MAC address.</p> <p>When the enable ARP response feature is activated, the X710-TM4 uses the following registers to filter in ARP requests. MC should not modify these registers:</p> <ul style="list-style-type: none"> • Manageability Decision Filter – MDEF7 (and corresponding bit 7 in Management Only traffic register – <i>MNGONLY</i>). • Fourth IPv4 Filter.
NM	5:4	<p>Notification Method. Define the notification method the X710-TM4 uses.</p> <p>00b = SMBUS alert.</p> <p>01b = Asynchronous notify.</p> <p>10b = Direct receive.</p> <p>11b = Not supported.</p> <p>Note: Changing the notification method in any port updates the notification method of all ports.</p>
Reserved	6	Reserved. Must be set to 1b.
CBDM	7	<p>Configure the MC Dedicated MAC Address.</p> <p>Note: This bit should be 0b when the <i>RCV_EN</i> bit (bit 0) is not set.</p> <p>0b = The X710-TM4 shares the MAC address for MNG traffic with the host MAC address, which is specified in NVM words 0x0-0x2. The MAC filtering is not enforced. See the note at the bottom of this table.</p> <p>1b = The X710-TM4 uses the MC dedicated MAC address as a filter for incoming receive packets. The MC MAC address is set in bytes 2-7 in this command.</p> <p>If a short version of the command is used, the X710-TM4 uses the MAC address configured in the most recent long version of the command in which the <i>CBDM</i> bit was set.</p> <p>When the dedicated MAC address feature is activated, the X710-TM4 uses the following registers to filter in all the traffic addressed to the MC MAC. MC can not modify these registers:</p> <p>Manageability Decision Filter – MDEF7 (and corresponding bit 7 in Management Only traffic register – <i>PRT_MNG_MNGONLY</i>).</p> <p>Manageability Decision Filter – MDEF6 (and corresponding bit 6 in Management Only traffic register – <i>MNGONLY</i>).</p> <p>Manageability MAC Address Low – <i>PRT_MNG_MMAL[3]</i>.</p> <p>Manageability MAC Address High – <i>PRT_MNG_MMAH[3]</i>.</p> <p>Note: When the dedicated MAC address feature is cleared, these registers are not programmed and the MC might use other filters to enforce MAC filtering using the Update Management Receive Filter Parameters command.</p>

9.5.10.1.4 Management MAC Address (Data Bytes 7:2)

Ignored if the *CBDM* bit is not set. This MAC address is used to configure the dedicated MAC address. In addition, it is used in the ARP response packet when the *EN_ARP_RES* bit is set. This MAC address is also used when *CBDM* bit is set in subsequent short versions of this command.



9.5.10.1.5 Management IP Address (Data Bytes 11:8)

This IP address is used to filter ARP request packets.

9.5.10.1.6 Asynchronous Notification SMBus Address (Data Byte 12)

This address is used for the asynchronous notification SMBus transaction and for direct receive. The SMBus address is stored in bit 7:1 of this byte. Bit 0 is always 0b.

9.5.10.1.7 Interface Data (Data Byte 13)

Interface data byte used in asynchronous notification.

9.5.10.1.8 Alert Value Data (Data Byte 14)

Alert value data byte used in asynchronous notification.

9.5.10.2 Force TCO Command

This command causes the X710-TM4 to perform a TCO reset, TCO isolate, or firmware reset

TCO reset — If force TCO reset is enabled in the NVM (see [Section 7.2.31.2](#)), the force TCO reset clears the data path (Rx/Tx) of the X710-TM4 to enable the MC to transmit/receive packets through the X710-TM4 by asserting a global reset. This command should only be used when the MC is unable to transmit receive and suspects that the X710-TM4 is inoperable. The command also causes the LAN software device driver to unload. It is recommended to perform a system restart to resume normal operation.

TCO isolate — if TCO isolate is enabled in the NVM (See [Section 7.2.31.3](#)), the TCO Isolate command disables PCIe write operations to the LAN port. If TCO isolate is disabled in NVM, the X710-TM4 does not execute the command but sends a response to the MC with successful completion. Following a TCO isolate, management sets *EMP_TCO_ISOLATE.EMP_TCO_ISOLATE* to 1b for all the PFs associated with the port on which this command is received. Once TCO isolate is set, the software device driver needs to be disabled, and is reported as such to the MC.

Firmware reset — This command causes re-initialization of all the embedded controller functions and re-load of related NVM words (like a firmware patch code). Applying this command resets the entire device as well as having an effect on TCO reset. A firmware reset is achieved by setting the *GSCR.SET_FWRST* aux bit.

Note: A firmware reset causes a global reset of the entire device (GLOBR).

The X710-TM4 considers the Force TCO Reset command as an indication that the operating system is unavailable. The Force TCO command format is as follows:

Function	Command	Byte Count	Data 1
Force TCO Reset	0xCF	1	TCO Mode



Where TCO mode is:

Field	Bit(s)	Description
DO_TCO_RST	0	Perform TCO Reset. 0b = Do nothing. 1b = Perform TCO reset.
DO_TCO_ISOLATE ¹	1	Do TCO Isolate. 0b = Enable PCIe write access to LAN port. 1b = Isolate Host PCIe write operation to the port. Note: Should be used for debug only.
RESET_MGMT	2	Reset Manageability; re-load manageability NVM words. 0b = Do nothing. 1b = Issue firmware reset to manageability. Setting this bit generates a one-time firmware reset. Following the reset, management related data from NVM is loaded.
Reserved	7:3	Reserved (set to 0x00).

1. TCO isolate host write operation enabled in NVM.

Note: Only one of the fields should be set in a given command. Setting more than one field might yield unexpected results.

9.5.10.2.1 Management Control

This command is used to set generic manageability parameters. The parameters are listed in [Table 9-11](#). The command is 0xC1 stating that it is a Management Control command. The first data byte is the parameter number and the data afterwards (length and content) are parameter specific as shown in Management Control Command Parameters/Content.

Note: If the parameter that the MC sets is not supported by the X710-TM4. The X710-TM4 does not NACK the transaction. After the transaction ends, the X710-TM4 discards the data and asserts a transaction abort status.

The Management Control command format is as follows:

Function	Command	Byte Count	Data 1	Data 2	...	Data N
Management Control	0xC1	N	Parameter Number	Parameter Dependent		



Table 9-11. Management Control Command Parameters/Content

Parameter	#	Parameter Data
Keep PHY Link Up	0x00	A single byte parameter: Data 2: Bit 0 = Set to indicate that the PHY link for this port should be kept up throughout system resets. This is useful when the server is reset and the MC needs to keep connectivity for a manageability session. Bit [7:1] = Reserved. 0b = Disabled. 1b = Enabled.

9.5.10.2.2 Update Management Receive Filter Parameters

This command is used to set the manageability receive filters parameters. The command is 0xCC. The first data byte is the parameter number and the data that follows (length and content) are parameter specific as listed in management RCV filter parameters.

If the parameter that the MC sets is not supported by the X710-TM4, then the X710-TM4 does not NACK the transaction. After the transaction ends, the X710-TM4 discards the data and asserts a transaction abort status.

The update management RCV receive filter parameters command format is as follows:

Function	Command	Byte Count	Data 1	Data 2	...	Data N
Update Manageability Filter Parameters	0xCC	N	Parameter Number	Parameter Dependent		

Table 9-12 lists the different parameters and their content.

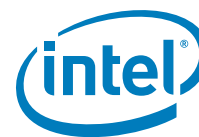
Table 9-12. Management Receive Filter Parameters

Parameter	Number	Parameter Data
Filters Enables	0x1	Defines the generic filters configuration. The structure of this parameter is four bytes as the Manageability Control (<i>PRT_MNG_MANC</i>) register. Note: The general filter enable is in the Receive Enable command that enables receive filtering.
MNGONLY configuration	0xF	This parameter defines which of the packets types identified as manageability packets in the receive path will never be directed to the host memory. Data 2:5 = <i>PRT_MNG_MNGONLY</i> register bytes - Data 2 is the MSB.
Flex Filter 0 Enable Mask and Length	0x10	Flex Filter 0 Mask. Data 17:2 = Mask. Bit 0 in data 2 is the first bit of the mask. Data 19:18 = Reserved. Should be set to 00b. Data 20 = Flexible filter length.



Table 9-12. Management Receive Filter Parameters

Parameter	Number	Parameter Data
Flex Filter 0 Data	0x11	Data 2 – Group of flex filter’s bytes: 0x0 = bytes 0-29. 0x1 = bytes 30-59. 0x2 = bytes 60-89. 0x3 = bytes 90-119. 0x4 = bytes 120-127. Data 3:32 = Flex filter data bytes. Data 3 is LSB. Group’s length is not a mandatory 30 bytes; it might vary according to filter’s length and must NOT be padded by zeros.
Decision Filters	0x61	This command is obsolete and should not be used. Please use 0x68 instead.
VLAN Filters	0x62	Three bytes are required to load the VLAN tag filters. Data 2: VLAN filter number. Data 3: MSB of VLAN filter. Data 4: LSB of VLAN filter.
Flex Port Filters	0x63	Three to four bytes are required to load the manageability flex port filters. Data 2 = Flex port filter number. Data 3 = MSB of flex port filter. Data 4 = LSB of flex port filter. Data 5: Bit 0 = Match UDP ports. Bit 1 = Match TCP ports. Bit 2 = Match destination port (0) or source port (1). If Data 5 is not present, the match is done on TCP and UDP destination ports (legacy behavior).
IPv4 Filters	0x64	Five bytes are required to load the IPv4 address filter. Data 2 = IPv4 address filter number (3:0). Data 3 = LSB of IPv4 address filter. ... Data 6 = MSB of IPv4 address filter.
IPv6 Filters	0x65	17 bytes are required to load the IPv6 address filter. Data 2 = IPv6 address filter number (3:0). Data 3 = LSB of IPv6 address filter. ... Data 18 = MSB of IPv6 address filter.
MAC Filters	0x66	Seven bytes are required to load the MAC address filters. Data 2 = MAC address filters pair number (3:0). Data 3 = MSB of MAC address. ... Data 8 = LSB of MAC address.

**Table 9-12. Management Receive Filter Parameters**

Parameter	Number	Parameter Data
EtherType Filters	0x67	Five bytes to load Ethertype Filters (METF). Data 2 = METF filter index (valid values are 0, 1, 2, 3). Data 3 = MSB of METF. ... Data 6 = LSB of METF.
Extended Decision Filter	0x68	Nine bytes to load the extended decision filters (MDEF_EXT & MDEF). Data 2 = MDEF filter index (valid values are 0..6). Data 3 – MSB of MDEF_EXT (DecisionFilter1). Data 6 = LSB of MDEF_EXT (DecisionFilter1). Data 7 = MSB of MDEF (DecisionFilter0). Data 10 = LSB of MDEF (DecisionFilter0). The command overwrites any previously stored value.
Management Special Filter Modifiers	0x69	Four bytes to load the Management Special Filter Modifiers. Data 2 = MSB of MSFM register. ... Data 5 = LSB of MSFM register.

Table 9-13. Filter Enable Parameters

Bit	Name	Description
16:0	Reserved	Reserved.
17	RCV_TCO_EN	Receive TCO Packets Enabled. When this bit is set it enables the receive flow to the manageability block. This bit should be set only if at least one of EN_BMC2OS or EN_BMC2NET bits are set. This bit is usually set using the receive enable command (see Section 9.5.10.1.3).
18	KEEP_PHY_LINK_UP	Block PHY reset and power state changes. When this bit is set the PHY reset and power state changes does not get to the PHY.
22:20	Reserved	Reserved.
23	Enable Xsum Filtering to MNG	When this bit is set, only packets that pass the L3 and L4 checksum are send to the manageability block.
24	Reserved	Reserved.
25	FIXED_NET_TYPE	Fixed Net Type. If set, only packets matching the net type defined by the <i>NET_TYPE</i> field passes to manageability. Otherwise, both tagged and un-tagged packets can be forwarded to the manageability engine.
26	NET_TYPE	Net Type. 0b = Pass only un-tagged packets. 1b = Pass only VLAN tagged packets. Valid only if <i>FIXED_NET_TYPE</i> is set.
31:27	Reserved	Reserved.

9.5.10.2.3 Set Common Filters Command

The Set Common Filters command is a single fragment command capable of configuring the most common filters.



Note: If this command is used, all the other commands that programs forwarding filters should not be used (apart from the Clear All Filters command). When this command is received, an implied Clear All Filters command is done before the application of this command.

The Set Common Filters command has two possible formats:

IPv4 format:

Function	Command	Byte Count	Data 1	Data 2:4	5:10	Data 11	Data 12	Data 13	Data 14:17
Set Common Filters	0xC2	17	Opcode = 0	Receive Control - see Table 9-14	MAC Address	BMC Alert Address	Interface Data Byte	Alert Value Byte	IPv4 Address

IPv6 format:

Function	Command	Byte Count	Data 1	Data 2:4	5:10	Data 11	Data 12	Data 13	Data 14:29
Set Common Filters	0xC2	29	Opcode = 0	Receive Control - see Table 9-14	MAC Address	BMC Alert Address	Interface Data Byte	Alert Value Byte	IPv6 Address



Table 9-14. Set Common Filters Receive Control Bytes

Byte	Bit	Field	Description
1	0	RCV_EN	Receive TCO Packets Enabled. When this bit is set it enables the receive flow to the manageability block. This bit should be set only if at least one of EN_BMC20 or EN_BMC2NET bits are set.
	1	EN_STA	Enable Status Reporting. 0b = Disable status reporting. 1b = Enable status reporting.
	2	Auto ARP	Automatically respond to ARP packets. Ignored in IPv6 mode. If this bit is set, broadcast ARP packets are handled by the X710-TM4 and ARP requests to the IP address set in the command are responded to. Notes: Mutually exclusive to Configure ARP/ Neighborhood Filter bit. If this bit is set, the IP address must be valid. This bit is ignored if RCV_EN is cleared.
	3	Enable Xsum Filtering to MNG	When this bit is set, only packets that pass the L3 and L4 checksum are send to the manageability block. This bit is ignored if RCV_EN is cleared.
	5:4	Reserved	Reserved.
	7:6	Notification Method	Notification Method. Define the notification method the X710-TM4 uses. 00b = SMBus alert. 01b = Asynchronous notify. 10b = Direct receive. 11b = Not supported.



Table 9-14. Set Common Filters Receive Control Bytes

Byte	Bit	Field	Description
2	8	CBDM	Configure the BMC Dedicated MAC Address. 0b = The X710-TM4 shares the MAC address for manageability traffic with the host MAC address, which is specified in NVM words 0x0-0x2. 1b = The X710-TM4 uses the MC dedicated MAC address as a filter for incoming receive packets. The MC MAC address is set in bytes 5:1 in this command. This bit is ignored if RCV_EN is cleared.
	9	Configure IP Address Filter	Automatically configure an IP address filter. If this bit is set, only packets matching this IP address is forwarded. If the <i>CBDM</i> bit is set, only packets matching the MAC and IP address is forwarded. This bit is ignored if RCV_EN is cleared.
	10	Configure RMCP 26Fh Filter	Automatically configure standard IPMI port 0x26F filters. If this bit is set, only packets matching this port is forwarded. If the <i>CBDM/Configure IP Address Filter</i> bits are set, only packets matching the MAC and IP address and this port is forwarded. The other port enable bit (11) might add additional forwarding conditions. This bit is ignored if RCV_EN is cleared.
	11	Configure RMCP 298h Filter	Automatically configure standard IPMI port 0x298 filter. If this bit is set, only packets matching this port is forwarded. If the <i>CBDM/Configure IP Address Filter</i> bits are set, only packets matching the MAC and IP address and this port is forwarded. The other port enable bit (10) might add additional forwarding conditions. This bit is ignored if RCV_EN is cleared.
	12	Configure ARP/ Neighborhood Filter	Automatically configure filters to enable this traffic to the MC (mutually exclusive to <i>Auto ARP</i> bit). If this bit is set, broadcast ARP packets are forwarded to the MC. In IPv4 mode, setting this bit enables forwarding of broadcast ARP requests and responses and unicast ARP responses. If the IP address is set, only a response to this address is forwarded. In IPv6 mode, setting this bit enables forwarding of all types of neighbor discovery and MLD ICMPv6 packet types: <ul style="list-style-type: none"> • 0x86 (134d) = Router Advertisement. • 0x87 (135d) = Neighbor Solicitation. • 0x88 (136d) = Neighbor Advertisement. • 0x89 (137d) = Redirect. • 0x82 (130d) = MLD Query. • 0x83 (131d) = MLDv1 Report. • 0x84 (132d) = MLD Done. • 0x8F (143d) = MLDv2 Report.
	13	Configure DHCP port 44h Filter	Automatically configure DHCP port 44 filter to the MC. If this bit is set, multicast packets matching this port is forwarded. Otherwise, multicast packets are not forwarded to the MC. This bit is ignored if RCV_EN is cleared or in IPv6 mode.
	15:14	Reserved	Reserved.
3	16	Disable Host ARP	Configure ARP requests and network neighborhood packets not to go to the host. This bit should be cleared during normal operation. Ignored if both bit 12 and bit 2 are cleared or if RCV_EN is cleared.
	17	Disable Host DHCP	Configure DHCP packets (port 0x44) not to go to host. This bit should be cleared in normal operation. Ignored if bit 13 is cleared, RCV_EN is cleared, or in IPv6 mode.
	24:18	Reserved	Reserved.

9.5.10.2.4 Clear All Filters Command

The Clear All Filters command is a single fragment command capable of clearing all the receive filters currently programmed for manageability traffic.



Function	Command	Byte Count	Data
Clear all Filters	0xC3	1	0x00

9.5.10.3 Read SMBus Transactions

This section details the Pass Through read transactions that the MC can send to the X710-TM4 over the SMBus.

SMBus read transactions lists the different SMBus read transactions supported by the X710-TM4. All the read transactions are compatible with SMBus read block protocol format.

Table 9-15. SMBus Read Transactions

TCO Command	Transaction	Command	Opcode	Fragments	Section
Receive TCO Packet	Block Read	0xD0 or 0xC0	First: 0x90 Middle: 0x10 Last ¹ : 0x50	Multiple	9.5.10.3.1
Read Status	Block Read	0xD0 or 0xC0 or 0xDE	Single: 0xDD	Single	9.5.10.3.2
Get System MAC Address	Block Read	0xD4	Single: 0xD4	Single	9.5.10.3.3
Read Management Parameters	Block Read	0xD1	Single: 0xD1	Single	9.5.10.3.4
Read Management RCV Filter Parameters	Block Read	0xCD	Single: 0xCD	Single	9.5.10.3.5
Read Receive Enable Configuration	Block Read	0xDA	Single: 0xDA	Single	9.5.10.3.7.1
Get Controller Information	Block Read	0xD5	Single: 0xD5	Single	9.5.10.3.6
Get Common Filters	Block Read	0xD3	Single: 0xD3	Single	9.5.10.3.7

1. The last fragment of the receive TCO packet is the packet status.

0xC0 or 0xD0 commands are used for more than one payload. If the MC issues these read commands, and the X710-TM4 has no pending data to transfer, it always returns as default opcode 0xDD with the X710-TM4 status and does not NACK the transaction.

If an SMBus Quick Read command is received, it is handled as a X710-TM4 Request Status command (see [Section 9.5.10.1.2](#) for details).

9.5.10.3.1 Receive TCO LAN Packet Transaction

The MC uses this command to read packets received on the LAN and its status. When the X710-TM4 has a packet to deliver to the MC, it asserts the SMBus notification for the MC to read the data (or direct receive). Upon receiving notification of the arrival of a LAN receive packet, the MC begins issuing a Receive TCO packet command using the block read protocol.

A packet can be transmitted to the MC in at least two fragments (at least one for the packet data and one for the packet status). As a result, the MC should follow the *F* and *L* bit of the opcode.

The opcode can have these values:



- 0x90 — First fragment
- 0x10 — Middle fragment
- When the opcode is 0x50, this indicates the last fragment of the packet, which contains packet status.

If a notification timeout is defined (in the NVM) and the MC does not finish reading the entire packet within the timeout period, since the packet has arrived, the packet is silently discarded. The time spent in ARA cycle or in reading the packet is not counted by the timeout counter.

Following is the receive TCO packet format and the data format returned from the X710-TM4.

Function	Command
Receive TCO Packet	0xC0 or 0xD0

Function	Byte Count	Data 1 (Opcode)	Data 2	...	Data N
Receive TCO First Fragment	N	0x90	Packet Data Byte	...	Packet Data Byte
Receive TCO Middle Fragment		0x10			
Receive TCO Last Fragment	9 (0x9)	0x50			

9.5.10.3.1.1 Receive TCO LAN Status Payload Transaction

This transaction is the last transaction that the X710-TM4 issues when a packet received from the LAN is transferred to the MC. The transaction contains the status of the received packet.

The format of the status transaction is as follows:

Function	Byte Count	Data 1 (Opcode)	Data 2 – Data 9 (Status Data)
Receive TCO Long Status	9 (0x9)	0x50	See Below

The status is 8 bytes where byte 0 (bits 7:0) is set in Data 2 of the status and byte 7 in Data 9 of the status. [Table 9-16](#) lists the content of the status data.

Table 9-16. TCO LAN Packet Status Data

Name	Bits	Description
Packet Length	13:0	Packet length including CRC, only 14 LSB bits.
Reserved	15:14	Reserved.
Packet status	31:16	See Table 9-17 .
Reserved	47:32	Reserved.
MNG status	63:48	See Table 9-19 . This field should be ignored if Receive TCO is not enabled.

**Table 9-17. Packet Status Info**

Field	Bit(s)	Description
Reserved	15:4	Reserved.
LAN#	3:2	Indicates the source port of the packet.
Reserved	1	Reserved.
CRC stripped	0	Insertion of CRC is needed.

Table 9-19. MNG Status

Name	Bits	Description
Reserved	15:9	Reserved.
Decision Filter match	8	Set when there is a match to one of the decision filters.
Decision Filter index	7:4	Indicates which of the decision filters match the packet. (allows for up to 16 filters - although only 8 are currently supported).
MNG VLAN Address Match	3	Set when the manageability packet matches one of the manageability VLAN filters.
Pass MNG VLAN Filter Index	2:0	Indicates which of the VLAN filters match the packet.

9.5.10.3.2 Read Status Command

The MC should use this command after receiving a notification from the X710-TM4 (such as SMBus alert). The X710-TM4 also sends a notification to the MC in either of the following two cases:

- The MC asserts a request for reading the status.
- The X710-TM4 detects a change in one of the Status Data 1 bits or NVM error bit in Data 2 (and was set to send status to the MC on status change) in the Receive Enable command.

Note: Commands 0xC0/0xD0 are for backward compatibility and can be used for other payloads. The X710-TM4 defines these commands in the opcode as well as which payload this transaction is. When the 0XDE command is set, the X710-TM4 always returns opcode 0XDD with the X710-TM4 status. The MC reads the event causing the notification, using the Read Status command as follows.

The X710-TM4's response to one of the commands (0xC0 or 0xD0) in a given time as defined in the SMBus Notification Timeout and Flags word in the NVM.

Function	Command
Read Status	0XC0 or 0XD0 or 0XDE



Function	Byte Count	Data 1 (Opcode)	Data 2 (Status Data 1)	Data 3 (Status Data 2)
Receive TCO Partial Status	3	0XDD	See Below	

This command can also be executed using the I²C quick read format as follows:

1	7	1	1	8	1	8	1	8	1	1
Start	Slave Address	Rd	Ack	Byte Count	Ack	Status Data 1	Ack	Status Data 2	Ack	Stop
		1	0	0000 0002	0		0		1	

Table 9-20 lists the status data byte 1 parameters.

Table 9-20. Status Data Byte 1

Bit	Name	Description															
7	LAN Port LSB	LAN port LSB together with LAN Port MSB defines port that sent status. See further information in the description of LAN Port MSB (bit 2).															
6	TCO Command Aborted	1b = A TCO command abort event occurred since the last read status cycle. 0b = A TCO command abort event did not occur since the last read status cycle.															
5	Link Status Indication	0b = LAN link down. 1b = LAN link up.															
4	PHY Link Forced Up	Contains the value of the <i>PHY_Link_Up</i> bit. When set, indicates that the PHY link is configured to keep the link up.															
3	Initialization Indication	0b = An NVM reload event has not occurred since the last read status cycle. 1b = An NVM reload event has occurred since the last read status cycle ¹ .															
2	LAN Port MSB	Defines together with LAN Port LSB the port that sent the status: <table border="0" style="margin-left: 20px;"> <tr> <td>Lan Port MSB</td> <td>Lan Port LSB</td> <td></td> </tr> <tr> <td>0</td> <td>0</td> <td>Status came from LAN port 0.</td> </tr> <tr> <td>0</td> <td>1</td> <td>Status came from LAN port 1.</td> </tr> <tr> <td>1</td> <td>0</td> <td>Status came from LAN port 2.</td> </tr> <tr> <td>1</td> <td>1</td> <td>Status came from LAN port 3.</td> </tr> </table>	Lan Port MSB	Lan Port LSB		0	0	Status came from LAN port 0.	0	1	Status came from LAN port 1.	1	0	Status came from LAN port 2.	1	1	Status came from LAN port 3.
Lan Port MSB	Lan Port LSB																
0	0	Status came from LAN port 0.															
0	1	Status came from LAN port 1.															
1	0	Status came from LAN port 2.															
1	1	Status came from LAN port 3.															
1:0	Power State	00b = Dr state. 01b = D0u state. 10b = D0 state. 11b = D3 state. Note: When more than one function is mapped to the same port, the highest power state of the mapped functions is reported according to the following order: Dr < D3 < D0u < D0.															

1. This indication is asserted when the X710-TM4 the manageability block reloads the NVM and its internal database is updated to the NVM default values. This is an indication that the external MC should reconfigure the X710-TM4, if other values other than the NVM default should be configured.

Status data byte 2 is used by the MC to indicate whether the LAN device driver is up and running.

The LAN device driver valid indication is a bit set by the LAN device driver during initialization; the bit is cleared when the LAN device driver enters a Dx state or is cleared by the hardware on a PCI reset.



Table 9-21 lists status data byte 2.

Table 9-21. Status Data Byte 2

Bit	Name	Description
7:6	Reserved	Reserved.
5	NVM Error	If set, indicates that a CRC/checksum error was detected in one of the manageability related NVM sections.
4	Reserved	Reserved.
3	Driver Valid Indication	0b = LAN driver is not up. 1b = LAN driver is up.
2:0	Reserved	Reserved.

9.5.10.3.3 Get System MAC Address

The get system MAC address returns the system MAC address over to the SMBus. This command is a single-fragment read block transaction that returns the system MAC address.

When a single function is defined on the port, it returns the LAN MAC address of this function as read from the PF allocations NVM section or from the alternate RAM or as set by the Manage MAC address write AQ command. When more than one function is defined on the port, it returns the address of the lowest defined function on this port.

Get system MAC address format:

Function	Command
Get system MAC address	0xD4

Data returned from the X710-TM4:

Function	Byte Count	Data 1 (Opcode)	Data 2	...	Data 7
Get system MAC address	7	0xD4	MAC address MSB	...	MAC address LSB

9.5.10.3.4 Read Management Parameters

In order to read the management parameters, the MC should execute two SMBus transactions. The first transaction is a block write that sets the parameter that the MC wants to read. The second transaction is block read that reads the parameter.

Block write transaction:

Function	Command	Byte Count	Data 1
Management control request	0xC1	1	Parameter number



Following the block write, the MC should issue a block read that reads the parameter that was set in the Block Write command:

Function	Command
Read management parameter	0xD1

Data returned:

Function	Byte Count	Data 1 (Opcode)	Data 2	Data 3	...	Data N
Read management parameter	N	0xD1	Parameter number	Parameter dependent		

The returned data is in the same format of the MC command.

The returned data is as follows:

Parameter	#	Parameter Data
Keep PHY Link Up	0x00	A single byte parameter: Data 2 – Bit 0 = Set to indicate that the PHY link for this port should be kept up. Sets the keep_PHY_link_up bit. When cleared, clears the keep_PHY_link_up bit. Bit [7:1] = Reserved.
Wrong parameter request	0xFE	Returned by the X710-TM4 only. This parameter is returned on a read transaction, if in the previous Read command the MC sets a parameter that is not supported by the X710-TM4.
X710-TM4 is not ready	0xFF	Returned by the X710-TM4 only, on a Read Parameters command when the data that should have been read is not ready. This parameter has no data. The MC should retry the read transaction. This value is also returned if the byte count is illegal or if the read command is not preceded by a Write command.

The parameter that is returned might not be the parameter requested by the MC. The MC should verify the parameter number (default parameter to be returned is 0x1).

If the parameter number is 0xFF, it means that the data that was requested from the X710-TM4 is not ready yet. The MC should retry the read transaction.

It is responsibility of the MC to follow the procedure previously defined. When the MC sends a Block Read command (as previously described) that is not preceded by a Block Write command with bytcount=1, the X710-TM4 sets the parameter number in the read block transaction to be 0xFF.

9.5.10.3.5 Read Management Receive Filter Parameters

In order to read the management receive filter parameters, the MC should execute two SMBus transactions. The first transaction is a block write that sets the parameter that the MC wants to read. The second transaction is block read that read the parameter.

Block write transaction:



Function	Command	Byte Count	Data 1	Data 2
Update MNG RCV filter parameters	0xCC	1 or 2	Parameter number	Parameter data

The different parameters supported for this command are the same as the parameters supported for the update management receive filter parameters.

Following the block write the MC should issue a block read that reads the parameter that was set in the Block Write command:

Function	Command
Request MNG RCV filter parameters	0xCD

Data returned from the X710-TM4:

Function	Byte Count	Data 1 (Opcode)	Data 2	Data 3	...	Data N
Read MNG RCV filter parameters	N	0xCD	Parameter number	Parameter dependent		

The parameter that is returned might not be the parameter requested by the MC. The MC should verify the parameter number (default parameter to be returned is 0x1).

If the parameter number is 0xFF, it means that the data that was requested from the X710-TM4 should supply is not ready yet. The MC should retry the read transaction.

It is MC's responsibility to follow the procedure previously defined. When the MC sends a Block Read command (as previously described) that is not preceded by a Block Write command with bytecount=1, the X710-TM4 sets the parameter number in the read block transaction to be 0xFF.

Parameter	#	Parameter Data
Filters Enable	0x01	None.
MNGONLY Configuration	0x0F	None.
Flex Filter Enable Mask and Length	0x10	None.
Flex Filter Data	0x11	Data 2 — Group of Flex Filter's Bytes: 0x0 = Bytes 0-29. 0x1 = Bytes 30-59. 0x2 = Bytes 60-89. 0x3 = Bytes 90-119. 0x4 = Bytes 120-127.
Filters Valid	0x60	None.
Decision Filters	0x61	This command is obsolete. Please use 0x68 instead.
VLAN Filters	0x62	One byte to define the accessed VLAN tag filter (PRT_MNG_MAVTV). Data 2 — VLAN Filter number.



Parameter	#	Parameter Data
Flex Ports Filters	0x63	One byte to define the accessed manageability flex port filter (PRT_MNG_MFUTP). Data 2 – Flex Port Filter number.
IPv4 Filter	0x64	One byte to define the accessed IPv4 address filter (PRT_MNG_MIPAF4). Data 2 – IPv4 address filter number.
IPv6 Filters	0x65	One byte to define the accessed IPv6 address filter (PRT_MNG_MIPAF6). Data 2 – Pv6 address filter number.
MAC Filters	0x66	One byte to define the accessed MAC address filters pair (PRT_MNG_MMAL, PRT_MNG_MMAH). Data 2 – MAC address filters pair number (0-3).
EtherType Filters	0x67	1 byte to define Ethertype filters (PRT_MNG_METF). Data 2 – METF filter index (valid values are 0 - 3).
Extended Decision Filter	0x68	1 byte to define the extended decisions filters (PRT_MNG_MDEF_EXT & PRT_MNG_MDEF). Data 2 – MDEF filter index (valid values are 0 - 6).
Management Special Filter Modifiers	0x69	
Wrong parameter request	0xFE	Returned by the X710-TM4 only. This parameter is returned on a read transaction, if in the previous Read command the MC sets a parameter that is not supported by the X710-TM4.
X710-TM4 is not ready	0xFF	Returned by the X710-TM4 only on a Read Parameters command when the data that should have been read is not ready. This parameter has no data. This value is also returned if the byte count is illegal or if the Read command is not preceded by a Write command.

9.5.10.3.6 Get Controller Information Command

The MC uses this command to get the controller identification. Each parameter is returned using a different parameter in the block write transaction.

In order to read the controller information, the MC should execute two SMBus transactions. The first transaction is a block write that sets the parameter that the MC wants to read. The second transaction is block read that read the parameter.

Block write transaction:

Function	Command	Byte Count	Data 1
Get Controller Information	0xD5	1	Parameter number

Following the block write, the MC should issue a block read that reads the parameter that was set in the Block Write command:

Function	Command
Get Controller Information	0xD5

Data returned from the X710-TM4:



Function	Byte Count	Command	Data 2 (Op-Code)	Data 3 -n
Get Controller Information	Per Table 9-22	0xD5	Per Table 9-22	See Table 9-22 for the data for each opcode

Table 9-22. Get Controller Information Data

Parameter	Byte Count	Description	Notes
0x00	5	Data 4:3: Device ID. Data 5: Silicon Revision (RevID).	This is the hardware default value, not any value programmed via the NVM.
0x0B	4	Data 4:3 NVM Image version.	
0x0C	6	Data 6:3: Firmware ROM Internal version.	
0x0D	6	Data 6:3: Firmware Flash Internal version.	
0x0E	4	Data 4:3: PXE FW version.	MajorVersion.MinorVersion.Build.SubBuild. If a version is not found, a value of 0xFFFF is returned.
0x0F	4	Data 4:3: iSCSI FW version.	
0x10	4	Data 4:3: uEFI FW version.	
0x16	4	Reserved.	
0xFE	2	Wrong parameter request.	Returned by the X710-TM4 only. This parameter is returned on a read transaction, if in the previous Read command the MC sets a parameter that is not supported by the X710-TM4, or if version of a non existing module is requested.
0xFF	2	X710-TM4 is not ready.	Returned by the X710-TM4 only, on a Read Parameters command when the data that should have been read is not ready. This parameter has no data. The MC should retry the read transaction. This value is also returned if the byte count is illegal or if the Read command is not preceded by a Write command.



9.5.10.3.7 Get Common Filters Command

The MC uses this command to get the common filters setting. This data can be configured when using Set Common Filters command. The first transaction is a block write that alerts that the MC needs to read the filters configuration. The second transaction is block read that read the configuration.

Block write transaction:

Function	Command	Byte count	Data
Get Common Filters	0xD3	1	0x00

Following the block write the MC should issue a block read that reads the filter settings:

Function	Command
Get Common filters	0xD3

Data returned from the X710-TM4:

Function	Byte Count	Command	Data 1	Data 2:4	5:10	Data 11	Data 12	Data 13	Data 14:17
Get Common Filters	18	0xD3	0	Receive Control - see Table 9-14	MAC Address	BMC Alert Address	Interface Data Byte	Alert Value Byte	IPv4 Address

Function	Byte Count	Command	Data 1	Data 2:4	5:10	Data 11	Data 12	Data 13	Data 14:29
Get Common Filters	30	0xD3	0	Receive Control - see Table 9-14	MAC Address	BMC Alert Address	Interface Data Byte	Alert Value Byte	IPv6 Address

If an error occurs, the following answers might be returned:

Function	Command	Byte Count	Data 1
Get Common Filters	0xD3	1	0xFF

This response is by the X710-TM4 on a Read Common Filter command when the data that should have been read is not ready. This parameter has no data. The MC should retry the read transaction.



This value is also returned if the byte count is illegal or if the Read command is not preceded by a Write command.

9.5.10.3.7.1 Read Receive Enable Configuration

The MC uses this command to read the receive configuration data. This data can be configured when using the Receive Enable command or through the NVM.

The Read Receive Enable Configuration command format (SMBus read block) is as follows:

Function	Command
Read Receive Enable	0xDA

Data returned from the X710-TM4:

Function	Byte Count	Data 1 (Opcode)	Data 2	Data 3	...	Data 8	Data 9	...	Data 12	Data 13	Data 14	Data 15
Read Receive Enable	15 (0x0F)	0xDA	Receive Control Byte	MAC Addr MSB	...	MAC Addr LSB	IP Addr MSB	...	IP Addr LSB	BMC SMBus Addr	I/F Data Byte	Alert Value Byte

The detailed description of each field is specified in the Receive Enable command description in [Section 9.5.10.1.3](#).

9.5.10.4 Example 1 - Shared MAC, RMCP Only Ports

This example is the most basic configuration. The MAC address filtering is shared with the host operating system and only traffic directed the RMCP ports (0x26F and 0x298) is filtered. For this example, the MC must issue gratuitous ARPs because no filter is enabled to pass ARP requests to the MC.

9.5.10.4.1 Example 1 Pseudo Code

Step 1: Disable existing filtering

Receive Enable[00]

Using the basic form of the Receive Enable command, this prevents any packets from reaching the MC by disabling filtering:

Receive Enable Control 0x00:

- Bit 0 [0] – Disable Receiving of packets

Step 2: Configure MDEF[0]

Update Manageability Filter Parameters [68, 0, C0000000, 00000000]

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 0x68). This updates MDEF[0], as indicated by the second parameter (0).

MDEF[0] value of 0xC0000000:



- Bit 30 [1] – port 0x298
 - Bit 31 [1] – port 0x26F
- MDEF_EXT[0] value of 0x00000000:

Step 3: Configure *MNGONLY*

Update Manageability Filter Parameters [F, 0, 00000001]

Use the Update Manageability Filter Parameters command to update Manageability Only (*MNGONLY*) (parameter 0xF) so that port 0x298 and 0x26F would not be sent to the host.

- Bit [0] - *MDEF[0]* is exclusive to the MC.

Step 4: - Enable Filtering

Receive Enable [05]

Using the basic form of the Receive Enable command:
 Receive Enable Control 0x05:

- Bit 0 [1] – Enable receiving of packets
- Bit 2 [1] – Enable status reporting (such as link lost)
- Bit 5:4 [00] – Notification method = SMB alert
- Bit 7 [0] – Use shared MAC

The resulting MDEF filters are as follows:

Table 9-23. Example 1 MDEF Results

Manageability Decision Filter (MDEF)									
Filter		0	1	2	3	4	5	6	7
L2 Unicast Address[3:0]	AND								
Broadcast	AND								
Manageability VLAN[7:0]	AND								
IPv6 Address[3:0]	AND								
IPv4 Address[3:0]	AND								
L2 Unicast Address[3:0]	OR								
Broadcast	OR								
Multicast	AND								
ARP Request	OR								
ARP Response	OR								
Neighbor Discovery	OR								
Port 0x298	OR	X							
Filter		0	1	2	3	4	5	6	7
Port 0x26F	OR	X							
Flex Port 7:0	OR								
Flex TCO	OR								



9.5.10.5 Example 2 - Dedicated MAC, Auto ARP Response and RMCP Port Filtering

This example shows a common configuration; the MC has a dedicated MAC and IP address. Automatic ARP responses are enabled as well as RMCP port filtering. By enabling automatic ARP responses, the MC is not required to send the gratuitous ARPs as it did in Example 1.

For demonstration purposes, the dedicated MAC address is calculated by reading the system MAC address and adding a one to it; assume the system MAC is AABBCDC. The IP address for this example is 1.2.3.4. Additionally, the XSUM filtering is enabled.

Note that not all Intel Ethernet controllers support automatic ARP responses, please refer to product specific documentation.

9.5.10.5.1 Example 2 - Pseudo Code

Step 1: Disable existing filtering

Receive Enable [00]

Using the basic form of the Receive Enable command, this prevents any packets from reaching the MC by disabling filtering:

Receive Enable Control 0x00:

- Bit 0 [0] – Disable Receiving of packets

Step 2: Read System MAC Address

Get System MAC Address []

Reads the system MAC address. Assume a returned AABBCDC for this example.

Step 3: Configure XSUM Filter

Update Manageability Filter Parameters [01, 00800000]

Use the Update Manageability Filter Parameters command to update Filters Enable settings (parameter 1). This sets the Manageability Control (*PRT_MNG_MANC*) register.

PRT_MNG_MANC Register 0x00800000:

- Bit 23 [1] - XSUM Filter Enable

Note that some of the following configuration steps manipulate the *PRT_MNG_MANC* register indirectly, this command sets all bits except XSUM to 0b. It is important to either do this step before the others, or to read the value of the *PRT_MNG_MANC* and then write it back with only bit 32 changed. Also note that the XSUM enable bit might differ between Ethernet controllers, refer to product specific documentation.

Step 4: Configure MDEF[0]

Update Manageability Filter Parameters [68, 0, C0000000, 00000000]

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 0x68). This updates MDEF[0], as indicated by the second parameter (0).

MDEF value of 0x00000C00:

- Bit 30 [1] – port 0x298
- Bit 31 [1] – port 0x26F

MDEF_EXT[0] value of 0x00000000:

Step 5: Configure MDEF[1]



Update Manageability Filter Parameters [68, 1, 10000000, 00000000]

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 0x61). This updates MDEF[1], as indicated by the second parameter (1).

MDEF value of 0x10000000:

- Bit 28 [1] – ARP Requests

MDEF_EXT[1] value of 0x00000000:

When enabling automatic ARP responses, the ARP requests still go into the manageability filtering system and as such need to be designated as also needing to be sent to the host. For this reason a separate MDEF is created with only ARP request filtering enabled.

Refer to the next step for more details.

Step 6: Configure Manageability only

Update Manageability Filter Parameters [F, 0, 00000001]

Use the Update Manageability Filter Parameters command to update Manageability Only (MNGONLY) (parameter 0xF) so that port 0x298 and 0x26F would not be sent to the host.

- Bit [0] - MDEF[0] is exclusive to the MC.

This enables ARP requests to be passed to both manageability and to the host. Specified separate MDEF filter for ARP requests. If ARP requests had been added to MDEF[0] and then MDEF[0] specified in management only configuration then not only would RMCP traffic (ports 0x26F and 0x298) be sent only to the MC, ARP requests would have also been sent to the MC only.

Step 7: Enable Filtering

Receive Enable [8D, AABBCDD, 01020304, 00, 00, 00]

Using the advanced version Receive Enable command, the first parameter:

Receive Enable Control 0x8D:

- Bit 0 [1] – Enable receiving of packets
- Bit 2 [1] – Enable status reporting (such as link lost)
- Bit 3 [1] – Enable automatic ARP responses
- Bit 5:4 [00] – Notification method = SMB alert
- Bit 7 [1] - Use dedicated MAC

Second parameter is the MAC address (AABBCDD).

Third parameter is the IP address(01020304).

The last three parameters are zero when the notification method is SMB alert.

The resulting MDEF filters are as follows:

Table 9-24. Example 2 MDEF Results

Manageability Decision Filter (MDEF)									
Filter		0	1	2	3	4	5	6	7
L2 Unicast Address[3:0]	AND								
Broadcast	AND								
Manageability VLAN[7:0]	AND								
IPv6 Address[3:0]	AND								
IPv4 Address[3:0]	AND								
L2 Unicast Address[3:0]	OR								



Table 9-24. Example 2 MDEF Results

Manageability Decision Filter (MDEF)									
Broadcast	OR								
Multicast	AND								
ARP Request	OR		X						
ARP Response	OR								
Neighbor Discovery	OR								
Port 0x298	OR	X							
Port 0x26F	OR	X							
Flex Port 7:0	OR								
Flex TCO	OR								

9.5.10.6 Example 3 - Dedicated MAC and IP Address

This example provided, the MC with a dedicated MAC and IP address enables it to receive ARP requests. The MC is then responsible for responding to ARP requests.

For demonstration purposes, the dedicated MAC address is calculated by reading the system MAC address and adding a one to it; assume the system MAC is AABBCDC. The IP address for this example is 1.2.3.4. For this example, the Receive Enable command is used to configure the MAC address filter.

In order for the MC to be able to receive ARP requests, it needs to specify a filter for this, and that filter needs to be included in the manageability-to-host filtering so that the host operating system can also receive ARP requests.

9.5.10.6.1 Example 3 - Pseudo Code

Step 1: Disable existing filtering

Receive Enable [00]

Using the basic form of the Receive Enable command, this prevents any packets from reaching the MC by disabling filtering:

Receive Enable Control 0x00:

- Bit 0 [0] – Disable receiving of packets

Step 2: Read System MAC Address

Get System MAC Address []

Reads the system MAC address. Assume a returned AABBCDC for this example.

Step 3: Configure IP Address Filter

Update Manageability Filter Parameters [64, 00, 01020304]

Use the update manageability filter parameters to configure an IPv4 filter.

The first parameter (0x64) specifies that we are configuring an IPv4 filter.

The second parameter (0x00) indicates which IPv4 filter is being configured; in this case filter 0.

The third parameter is the IP address – 1.2.3.4.

Step 4: Configure MAC Address Filter



Update Manageability Filter Parameters [66, 00, AABBCDD]

Use the update manageability filter parameters to configure a MAC address filter.

The first parameter (0x66) specifies that we are configuring a MAC address filter.

The second parameter (0x00) indicates which MAC address filter is being configured; in this case filter 0.

The third parameter is the MAC address - AABBCDD

Step 5: Configure MDEF[0] for IP and MAC Filtering

Update Manageability Filter Parameters [68, 0, 00002001, 00000000]

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 0x68). This updates MDEF[0], as indicated by the second parameter (0).

MDEF value of 00002001:

- Bit 0 [1] – MAC[0] address filtering
- Bit 13 [1] – IP[0] address filtering

MDEF_EXT[0] value of 0x00000000:

Step 6: Configure MDEF[1]

Update Manageability Filter Parameters [68, 1, 10000000]

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 0x68). This updates MDEF[1], as indicated by the second parameter (1).

MDEF value of 10000000:

- Bit 28 [1] – ARP requests

MDEF_EXT[1] value of 0x00000000:

Step 7: Configure the Management to Host Filter

Update Manageability Filter Parameters [F, 0, 00000001]

Use the Update Manageability Filter Parameters command to update Manageability Only (*MNGONLY*) (parameter 0xF) so that the dedicated MAC/IP traffic would not be sent to the host. Note that given the host does not program this address in its L2 filtering, this step is not a must, unless the host chooses to work in promiscuous mode.

- Bit [0] - MDEF[0] is exclusive to the MC.

Step 8: Enable Filtering

Receive Enable [05]

Using the basic form of the Receive Enable command,:

Receive Enable Control 0x05:

- Bit 0 [1] – Enable receiving of packets
- Bit 2 [1] – Enable status reporting (such as link lost)
- Bit 5:4 [00] – Notification method = SMB alert

The resulting MDEF filters are as follows:



Table 9-25. Example 3 MDEF Results

Manageability Decision Filter (MDEF)									
Filter		0	1	2	3	4	5	6	7
L2 Unicast Address[3:0]	AND	0001							
Broadcast	AND								
Manageability VLAN[7:0]	AND								
IPv6 Address[3:0]	AND								
IPv4 Address[3:0]	AND	0001							
L2 Unicast Address[3:0]	OR								
Broadcast	OR								
Multicast	AND								
ARP Request	OR		X						
ARP Response	OR								
Neighbor Discovery	OR								
Port 0x298	OR								
Port 0x26F	OR								
Flex Port 7:0	OR								
Flex TCO	OR								

9.5.10.7 Example 4 - Dedicated MAC and VLAN Tag

This example shows an alternate configuration; the MC has a dedicated MAC and IP address, along with a VLAN tag of 0x32 are required for traffic to be sent to the MC. This means that all traffic with a VLAN and matching tag is sent to the MC.

For demonstration purposes, the dedicated MAC address is calculated by reading the system MAC address and adding a one to it; assume the system MAC is AABBCDC. The IP address for this example is 1.2.3.4 and the VLAN tag is 0x0032.

Additionally, the XSUM filtering is enabled.

9.5.10.7.1 Example 4 - Pseudo Code

Step 1: Disable existing filtering

Receive Enable [00]

Using the basic form of the Receive Enable command, this prevents any packets from reaching the MC by disabling filtering:

Receive Enable Control 0x00:

- Bit 0 [0] – Disable receiving of packets

Step 2: - Read System MAC Address

Get System MAC Address []

Reads the system MAC address. Assume a returned AABBCDC for this example.



Step 3: Configure XSUM Filter

Update Manageability Filter Parameters [01, 00800000]

Use the Update Manageability Filter Parameters command to update filters enable settings (parameter 1). This sets the Manageability Control (*PRT_MNG_MANC*) register.

PRT_MNG_MANC Register 0x00800000:

- Bit 23 [1] – XSUM filter enable

Note that some of the following configuration steps manipulate the *PRT_MNG_MANC* register indirectly. This command sets all bits except XSUM to 0b. It is important to either do this step before the others, or to read the value of the *PRT_MNG_MANC* and then write it back with only bit 32 changed. Also note that the XSUM enable bit might differ between Ethernet controllers, refer to product specific documentation.

Step 4: Configure VLAN 0 Filter

Update Manageability Filter Parameters [62, 0, 0032]

Use the Update Manageability Filter Parameters command to configure VLAN filters. Parameter 0x62 indicates an update to the VLAN filter. The second parameter indicates which VLAN filter (0 in this case). The last parameter is the VLAN ID (0x0032).

Step 5: Configure MDEF[0]

Update Manageability Filter Parameters [68, 0, 00000020, 00000000]

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 0x68). This updates MDEF[0], as indicated by the second parameter (0).

MDEF value of 0x00000020:

- Bit 5 [1] – VLAN[0] AND
MDEF_EXT[0] value of 0x00000000:

Step 6: Enable Filtering

Receive Enable [85, AABBCDD, 01020304, 00, 00, 00]

Using the advanced version Receive Enable command, the first parameter:

Receive Enable Control 0x85:

- Bit 0 [1] – Enable receiving of packets
- Bit 2 [1] – Enable status reporting (such as link lost)
- Bit 5:4 [00] – Notification method = SMB alert
- Bit 7 [1] – Use dedicated MAC

Second parameter is the MAC address: AABBCDD.

The third parameter is the IP address: 01020304.

The last three parameters are zero when the notification method is SMBus alert.

The resulting MDEF filters are as follows:

Table 9-26. Example 4 MDEF Results

		Manageability Decision Filter (MDEF)							
Filter		0	1	2	3	4	5	6	7
L2 Unicast Address[3:0]	AND								0001
Broadcast	AND								



Table 9-26. Example 4 MDEF Results

Manageability Decision Filter (MDEF)									
Manageability VLAN[7:0]	AND	X							
IPv6 Address[3:0]	AND								
IPv4 Address[3:0]	AND								
L2 Unicast Address[3:0]	OR								
Broadcast	OR								
Multicast	AND								
ARP Request	OR								
ARP Response	OR								
Neighbor Discovery	OR								
Port 0x298	OR								
Port 0x26F	OR								
Flex Port 7:0	OR								
Flex TCO	OR								

9.5.11 SMBus Troubleshooting

This section outlines the most common issues found while working with PT using the SMBus sideband interface.

9.5.11.1 TCO Alert Line Stays Asserted After a Power Cycle

After the X710-TM4 resets, all its ports indicates a status change. If the MC only reads status from one port (slave address), the other one continues to assert the TCO alert line.

Ideally, the MC should use the ARA transaction to determine which slave asserted the TCO alert. Many customers only want to use one port for manageability, thus using ARA might not be optimal.

An alternative to using ARA is to configure part of the ports to not report status and to set its SMBus timeout period. In this case, the SMBus timeout period determines how long a port asserts the TCO alert line awaiting a status read from a MC; by default this value is zero (indicates an infinite timeout).

The SMBus configuration section of the NVM has a *SMBus Notification Timeout* (ms) field that can be set to a recommended value of 0xFF (for this issue). Note that this timeout value is for all slave addresses. Along with setting the *SMBus Notification Timeout* to 0xFF, it is recommended that the other ports be configured in the NVM to disable status alerting. This is accomplished by having the *Enable Status Reporting* bit set to 0b for the desired ports in the LAN configuration section of the NVM.

The third solution for this issue is to have the MC hard-code the slave addresses to always read from all ports. As with the previous solution, it is recommend that the other ports have status reporting disabled.



9.5.11.2 When SMBus Commands are Always NACK'd

There are several reasons why all commands sent to the X710-TM4 from a MC could be NACK'd. The following are most common:

- Invalid NVM Image — The image itself might be invalid or it could be a valid image and is not a PT image, as such SMBus connectivity is disabled.
- The MC is not using the correct SMBus address — Many MC vendors hard-code the SMBus address(es) into their firmware. If the incorrect values are hard-coded, the X710-TM4 does not respond.
 - The SMBus address(es) can be dynamically set using the SMBus ARP mechanism.
- The MC is using the incorrect SMBus interface — The NVM might be configured to use one physical SMBus port; however, the MC is physically connected to a different one.
- Bus Interference — The bus connecting the MC and the X710-TM4 might be unstable.

9.5.11.3 SMBus Clock Speed is 16.6666 KHz

This can happen when the SMBus connecting the MC and the X710-TM4 is also tied into another device (such as an ICH) that has a maximum clock speed of 16.6666 KHz. The solution is to not connect the SMBus between the X710-TM4 and the MC to this device.

9.5.11.4 A Network-based Host Application is not Receiving any Network Packets

Reports have been received about an application not receiving any network packets. The application in question was NFS under Linux. The problem was that the application was using the RMPC/RMCP+ IANA reserved port 0x26F (623) and the system was also configured for a shared MAC and IP address with the operating system and the MC.

The management control to host configuration, in this situation, was setup not to send RMCP traffic to the operating system (this is typically the correct configuration). This means that no traffic sent to port 623 was being routed.

The solution in this case is to configure the problematic application NOT to use the reserved port 0x26F.

9.5.11.5 Unable to Transmit Packets From the MC

If the MC has been transmitting and receiving data without issue for a period of time and then begins to receive NACKs from the X710-TM4 when it attempts to write a packet, the problem is most likely due to the fact that the buffers internal to the X710-TM4 are full of data that has been received from the network but has yet to be read by the MC.

Being an embedded device, the X710-TM4 has limited buffers that are shared for receiving and transmitting data. If a MC does not keep the incoming data read, the X710-TM4 can be filled up. This prevents the MC from transmitting more data, resulting in NACKs.

If this situation occurs, the recommended solution is to have the MC issue a Receive Enable command to disable more incoming data, read all the data from the X710-TM4, and then use the Receive Enable command to enable incoming data.



9.5.11.6 SMBus Fragment Size

The SMBus specification indicates a maximum SMBus transaction size of 32 bytes. Most of the data passed between the X710-TM4 and the MC over the SMBus is RMCP/RMCP+ traffic, which by its very nature (UDP traffic) is significantly larger than 32 bytes in length. Multiple SMBus transactions might therefore be required to move data from the X710-TM4 to the MC or to send a data from the MC to the X710-TM4.

Recognizing this bottleneck, the X710-TM4 handles up to 240 bytes of data in a single transaction. This is a configurable setting in the NVM. The default value in the NVM images is 32, per the SMBus specification. If performance is an issue, increase this size.

During initialization, firmware within the X710-TM4 allocates buffers based upon the SMBus fragment size setting within the NVM. X710-TM4 firmware has a finite amount of RAM for its use: the larger the SMBus fragment size, the fewer buffers it can allocate. Because this is true, MC implementations must take care to send data over the SMBus efficiently.

For example, the X710-TM4 firmware has 3 KB of RAM it can use for buffering SMBus fragments. If the SMBus fragment size is 32 bytes then the firmware could allocate 96 buffers of size 32 bytes each. As a result, the MC could then send a large packet of data (such as KVM) that is 800 bytes in size in 25 fragments of size 32 bytes apiece.

However, this might not be the most efficient way because the MC must break the 800 bytes of data into 25 fragments and send each one at a time.

If the SMBus fragment size is changed to 240 bytes, the X710-TM4 firmware can create 12 buffers of 240 bytes each to receive SMBus fragments. The MC can now send that same 800 bytes of KVM data in only four fragments, which is much more efficient.

The problem of changing the SMBus fragment size in the NVM is if the MC does not also reflect this change. If a programmer changes the SMBus fragment size in the X710-TM4 to 240 bytes and then wants to send 800 bytes of KVM data, the MC can still only send the data in 32 byte fragments. As a result, firmware runs out of memory.

This is because firmware created the 12 buffers of 240 bytes each for fragments; however, the MC is only sending fragments of size 32 bytes. This results in a memory waste of 208 bytes per fragment. Then when the MC attempts to send more than 12 fragments in a single transaction, the X710-TM4 NACKs the SMBus transaction due to not enough memory to store the KVM data.

In summary, if a programmer increases the size of the SMBus fragment size in the NVM (recommended for efficiency purposes) take care to ensure that the MC implementation reflects this change and uses that fragment size to its fullest when sending SMBus fragments.

9.5.11.7 Losing Link

Normal behavior for the Ethernet controller when the system powers down or performs a reset is for the link to temporarily go down and then back up again to re-negotiate the link speed. This behavior can have adverse affects on manageability.

For example, if there is an active FTP or Serial Over LAN (SoL) session to the MC, this connection can be lost. In order to avoid this possible situation, the MC can use the Management Control command to ensure the link stays active at all times.

This command is available when using the NC-SI sideband interface as well.



Care should be taken with this command, if the software device driver negotiates the maximum link speed, the link speed remains the same when the system powers down or resets. This can have undesirable power consumption consequences. Currently, when using NC-SI, the MC can re-negotiate the link speed. That functionality is not available when using the SMBus interface.

9.5.11.8 Enable Checksum Filtering

If checksum filtering is enabled, the MC does not need to perform the task of checking this checksum for incoming packets. Only packets that have a valid checksum is passed to the MC. All others are silently discarded.

This is a way to offload some work from the MC.

9.5.11.9 Still Having Problems?

If problems still exist, contact your field representative. Be prepared to provide the following:

- A SMBus trace if possible.
- A dump of the NVM image. This should be taken from the actual X710-TM4, rather than the NVM image provided by Intel. Parts of the NVM image are changed after writing (such as the physical NVM size).

9.6 Network Controller Sideband Interface (NC-SI) PT Interface

The NC-SI is a DMTF industry standard protocol for the sideband interface. NC-SI uses a modified version of the industry standard RMII interface for the physical layer (RBT) as well as defining a new logical layer.

The NC-SI specification can be found at:

<http://www.dmtf.org/>

9.6.1 Overview

9.6.1.1 Terminology

The terminology in this section is taken from the NC-SI specification.



Table 9-27. NC-SI Terminology

Term	Definition
Frame Versus Packet	Frame is used in reference to Ethernet, whereas packet is used everywhere else.
External Network Interface	The interface of the network controller that provides connectivity to the external network infrastructure (port).
Internal Host Interface	The interface of the network controller that provides connectivity to the host OS running on the platform.
Management Controller (BMC)	An intelligent entity comprising of hardware/firmware/software that resides within a platform and is responsible for some or all management functions associated with the platform (MC, service processor, etc.).
Network Controller (NC)	The component within a system that is responsible for providing connectivity to the external Ethernet network world.
Remote Media	The capability to allow remote media devices to appear as if they were attached locally to the host.
Network Controller Sideband Interface	The interface of the network controller that provides connectivity to a management controller. It can be shorten to sideband interface as appropriate in the context.
Interface	This refers to the entire physical interface, such as both the transmit and receive interface between the management controller and the network controller.
Integrated Controller	The term integrated controller refers to a network controller device that supports two or more channels for NC-SI that share a common NC-SI physical interface. For example, a network controller that has two or more physical network ports and a single NC-SI bus connection.
Multi-Drop	Multi-drop commonly refers to the case where multiple physical communication devices share an electrically common bus and a single device acts as the master of the bus and communicates with multiple slave or target devices. In NC-SI, a management controller serves the role as the master, and the network controllers are the target devices.
Point-to-Point	Point-to-point commonly refers to the case where only two physical communication devices are interconnected via a physical communication medium. The devices might be in a master/slave relationship, or could be peers. In NC-SI, point-to-point operation refers to the situation where only a single management controller and single network controller package are used on the bus in a master/slave relationship where the management controller is the master.
Channel	The control logic and data paths supporting NC-SI pass-through operation on a single network interface (port). A network controller that has multiple network interface ports can support an equivalent number of NC-SI channels.
Package	One or more NC-SI channels in a network controller that share a common set of electrical buffers and common buffer control for the NC-SI bus. Typically, there will be a single, logical NC-SI package for a single physical network controller package (chip or module). However, the specification allows a single physical chip or module to hold multiple NC-SI logical packages.
Control Traffic/Messages/Packets	Command, response and notification packets transmitted between BMC and the X710-TM4 for the purpose of managing NC-SI.
Pass-Through Traffic/Messages/Packets	Non-control packets passed between the external network and the BMC through the X710-TM4.
Channel Arbitration	Refer to operations where more than one of the network controller channels can be enabled to transmit pass-through packets to the MC at the same time, where arbitration of access to the RXD and CRS_DV signal lines is accomplished either by software or hardware means.
Logically Enabled/Disabled NC	Refers to the state of the network controller wherein pass-through traffic is able/unable to flow through the sideband interface to and from the management controller, as a result of issuing Enable/Disable Channel command.
NC RX	Defined as the direction of ingress traffic on the external network controller interface

Table 9-27. NC-SI Terminology

Term	Definition
NC TX	Defined as the direction of egress traffic on the external network controller interface
NC-SI RX	Defined as the direction of ingress traffic on the sideband enhanced NC-SI Interface with respect to the network controller.
NC-SI TX	Defined as the direction of egress traffic on the sideband enhanced NC-SI Interface with respect to the network controller.

9.6.1.2 System Topology

In NC-SI each physical endpoint (NC package) can have several logical slaves (NC channels).

NC-SI defines that one MC and up to four network controller packages can be connected to the same NC-SI link.

Figure 9-6 shows an example topology for a single MC and a single NC package. In this example, the NC package has two NC channels.

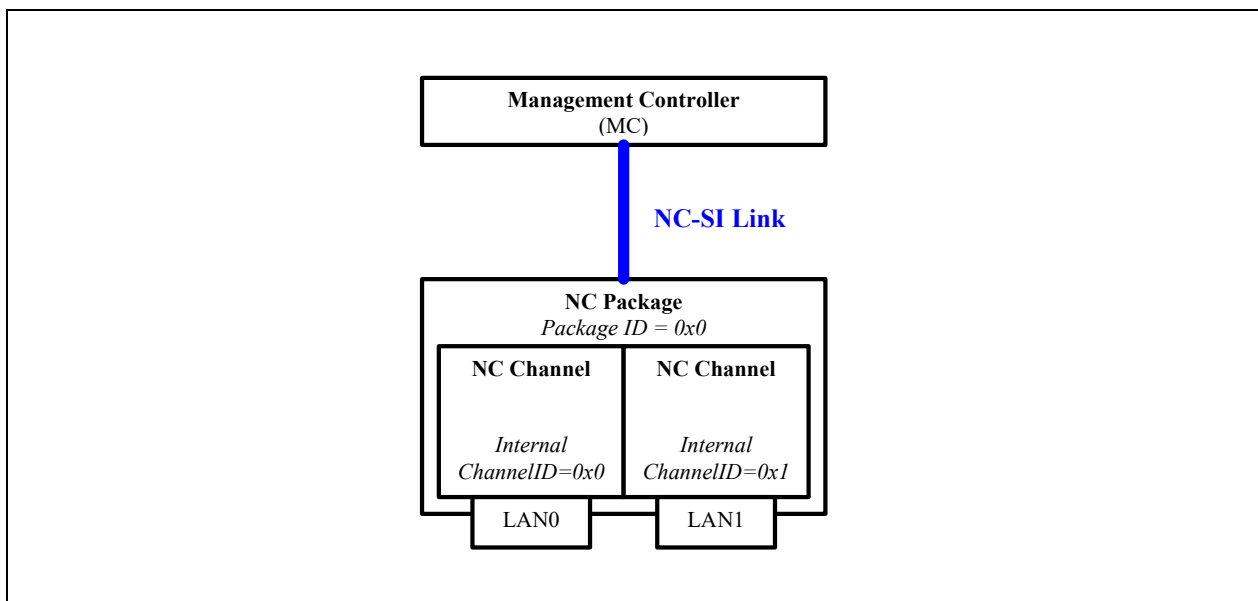


Figure 9-6. Single NC Package (Two NC Channels)

Figure 9-7 shows an example topology for a single MC and two NC packages. In this example, one NC package has two NC channels and the other has only one NC channel. Scenarios in which the NC-SI lines are shared by multiple NCs (Figure 9-7) mandate an arbitration mechanism.

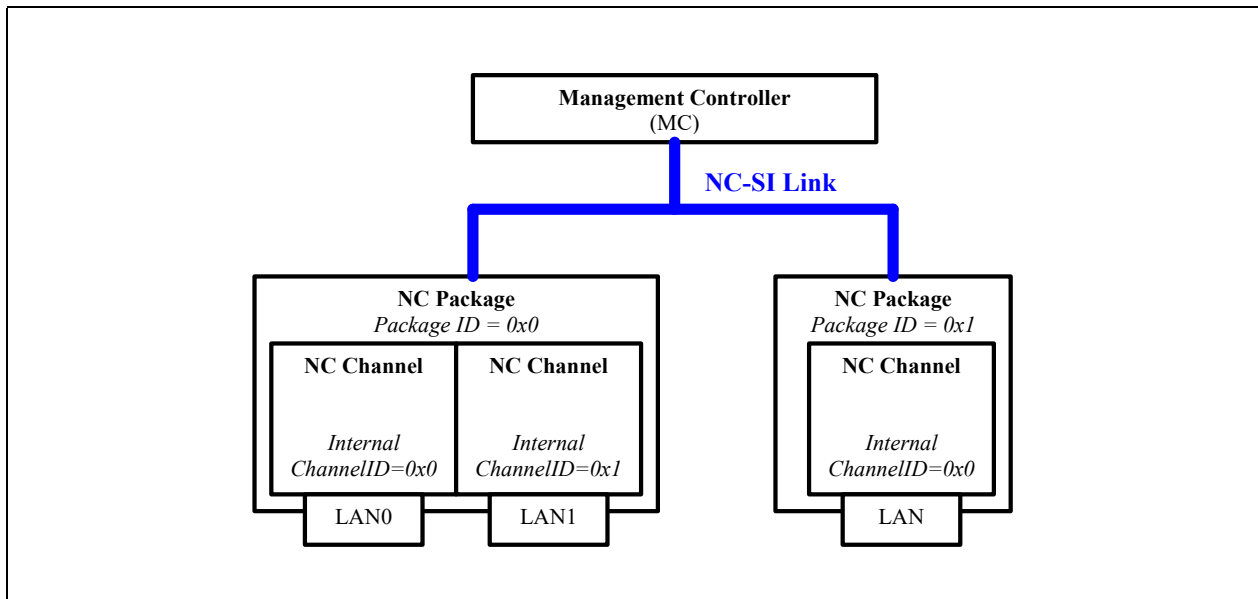


Figure 9-7. Two NC Packages - Left, Two NC Channels - Right, One NC Channel)

Note: Channel numbers should match PCI function numbers. If more than one function is defined on a port, the function with the lowest value associated with this port is used. See [Section 9.2.2.3.2](#) for details.

9.6.1.3 Data Transport

Since NC-SI is based upon the RMIi transport layer, data is transferred in the form of Ethernet frames.

NC-SI defines two types of transmitted frames:

1. Control frames:
 - a. Configures and control the interface.
 - b. Identified by a unique EtherType in their L2 header.
2. PT frames:
 - a. Actual LAN pass-through frames transferred from/to the MC.
 - b. Identified as not being a control frame.
 - c. Attributed to a specific NC channel by their source MAC address (as configured in the NC by the MC).

9.6.1.3.1 Control Frames

NC-SI control frames are identified by a unique NC-SI EtherType (0x88F8).

Control frames are used in a single-threaded operation, meaning commands are generated only by the MC and can only be sent one at a time. Each command from the MC is followed by a single response from the NC (command-response flow), after which the MC is allowed to send a new command.



The only exception to the command-response flow is the Asynchronous Event Notification (AEN). These control frames are sent unsolicited from the NC to the MC.

AEN functionality by the NC must be disabled by default, until activated by the MC using the AEN Enable commands.

In order to be considered a valid command, a control frame must:

1. Comply with the NC-SI header format.
2. Be targeted to a valid channel in the package via the *Package ID* and *Channel ID* fields. For example, to target a NC channel with package ID of 0x2 and internal channel ID of 0x5, the MC must set the channel ID inside the control frame to 0x45. The channel ID is composed of three bits of package ID and five bits of internal channel ID.
3. Contain a correct payload checksum (if used).
4. Meet any other condition defined by NC-SI.

There are also commands (such as select package) targeted to the package as a whole. These commands must use an internal channel ID of 0x1F.

For details, refer to the NC-SI specification.

9.6.1.3.2 NC-SI Frames Receive Flow

Figure 9-8 shows the flow for frames received on the NC from the MC.

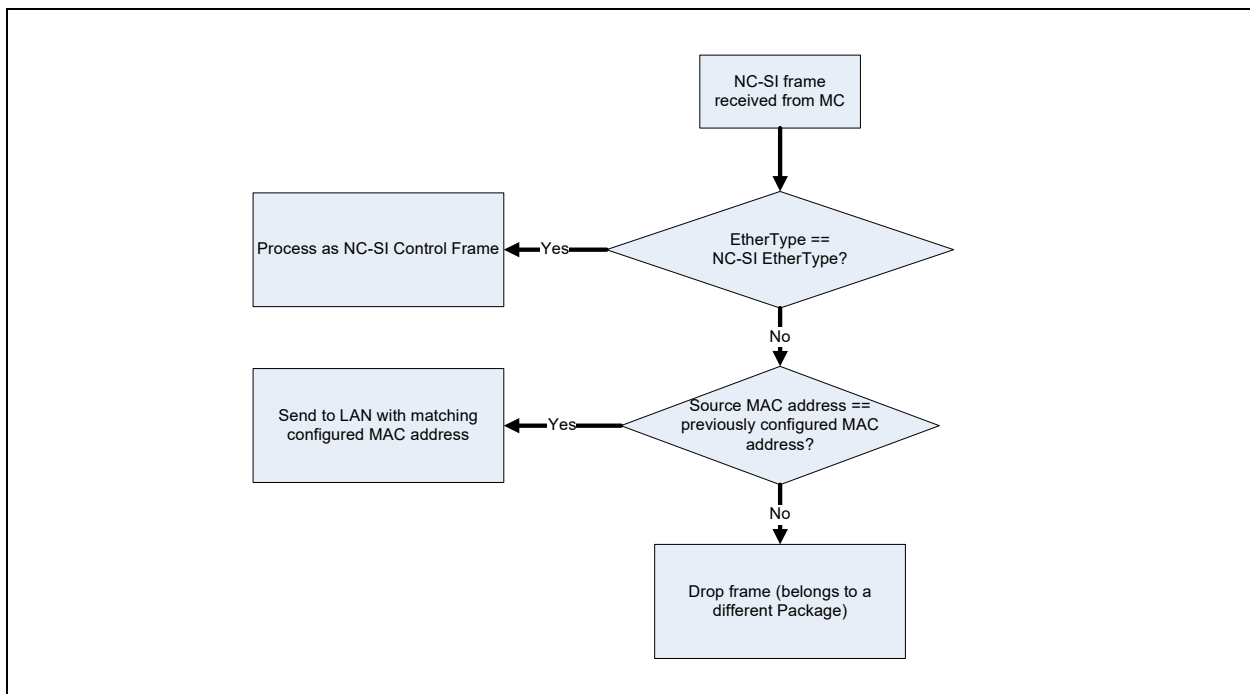


Figure 9-8. NC-SI Frames Receive Flow for NC



9.6.2 NC-SI Standard Support

9.6.2.1 Supported Features

The X710-TM4 supports all the mandatory features of the NC-SI specification. [Table 9-28](#) lists the support for standard NC-SI commands.

The X710-TM4 supports either NC-SI 1.0.1 or NC-SI 1.1.0 according to the NC-SI version flag in the NVM NC-SI Configuration 1 word.

[Table 9-29](#) lists optional features supported and the level of support for partially supported commands.

Table 9-28. NC-SI Commands Support

Command	Supported Over RMI	Supported Over MCTP With Pass Through	Supported Over MCTP Without Pass Through
Clear initial state	Yes	Yes	Yes
Get Version ID	Yes	Yes	Yes
Get Parameters	Yes	Yes	Yes
Get Controller Packet Statistics	Yes, partially	Yes, partially	Yes, partially
Get Link Status ¹	Yes	Yes	Yes
Enable Channel	Yes	Yes	Yes
Disable Channel	Yes	Yes	Yes
Reset Channel	Yes	Yes	Yes
Enable VLAN	Yes ^{2,3}	Yes ³	No ⁴
Disable VLAN	Yes	Yes	No ⁵
Enable Broadcast Filter	Yes	Yes	No ⁵
Disable Broadcast Filter	Yes	Yes	No ⁵
Set MAC Address	Yes	Yes	No ⁵
Get NC-SI Statistics	Yes	Yes	Yes
Set NC-SI Flow-Control	Yes	No	No ⁵
Set Link Command	Yes	Yes	Yes
Enable Global multicast Filter	Yes	Yes	No ⁵
Disable Global multicast Filter	Yes	Yes	No ⁵
Get Capabilities	Yes	Yes	Yes ⁵
Set VLAN Filters	Yes	Yes	No ⁵
AEN Enable	Yes	Yes	Yes
Get NC-SI Pass-Through Statistics	Yes, partially	Yes, partially	No ⁵
Select Package	Yes	Yes	Yes
Deselect Package	Yes	Yes	Yes



Table 9-28. NC-SI Commands Support (Continued)

Command	Supported Over RMI	Supported Over MCTP With Pass Through	Supported Over MCTP Without Pass Through
Enable Channel Network TX	Yes	Yes	No
Disable Channel Network TX	Yes	Yes	No
Get Package Status	Yes	No	No
Get Package UUID ⁶	Yes	Yes	Yes
PLDM Request	No	No	No
OEM Command	Yes	Yes	Yes

1. Parallel detection flag in this command response is not supported.
2. In cases that one of the LAN devices is assigned for the sole use of the manageability and its LAN PCIe function is disabled, using the NC-SI Set Link command while advertising multiple speeds and enabling auto-negotiation, results in the lowest possible speed chosen. To enable link of higher a speed, the MC should not advertise speeds that are below the desired link speed. When doing it, changing the power state of the LAN device has no effect and the link speed is not re-negotiated.
3. The X710-TM4 does not support filtering of *User Priority/CFI* bits of VLAN.
4. In MCTP without PT mode, only control commands are supported and not PT traffic. Thus many of the regular NC-SI commands are not supported or are supported in a limited manner, only to enable control and status reporting for the device.
5. When PT is disabled, the Get Capabilities command does not expose all the filtering capabilities of the device.
6. Only relevant in NC-SI 1.1

Table 9-29. Optional NC-SI Features Support

Feature	Implement	Details
AENs	Yes	The following AENs are supported: <ul style="list-style-type: none"> • Link Change AEN • Configuration Required AEN • Driver Status Change AEN • Initialization error AEN (Intel OEM AEN) The Driver state AEN might be emitted up to 1 minute after actual driver change if the driver was taken down unexpectedly.
Get Controller Packet Statistics command	Yes, partially	Supports the following counters ¹ : 0-8,11-16, 21-36 ² The statistics are not cleared between reads. Note: The packets counted by counter #35 (1523 - 9022 Byte Frames transmitted) are up to 9522 bytes and not 9022 as requested in the spec. If a core reset occurs between reads of statistics, the Counters Cleared From Last Read flags are set in the response.
Get NC-SI statistics	Yes	Support all the counters ³ Note: If a core reset occurs between reads of statistics, the reported statistics may be wrong.
Get NC-SI Pass-Through Statistics	Yes, partially	Support the following counters: 1, 6, 7.
VLAN Modes	Yes, partially	Support only modes 1, 3.
Buffering Capabilities	Yes	8 Kb
MAC Address Filters	Yes	Supports 2 MAC addresses per port.
Channel Count	Yes	Supports four channels.
VLAN Filters	Yes	Support 8 VLAN filters per port. Filtering is ignoring the <i>CFI</i> bit and the 802.1P priority bits

**Table 9-29. Optional NC-SI Features Support**

Feature	Implement	Details
Broadcast Filters	Yes	Support the following filters: ARP DHCP Net BIOS
Multicast Filters	Yes	Supports the following filters: <ul style="list-style-type: none"> • IPv6 neighbor advertisement • IPv6 router advertisement • DHCPv6 relay and server multicast In NC-SI 1.1 mode, the following filters are also supported <ul style="list-style-type: none"> • DHCPv6 multicasts from server to clients listening on well-known UDP ports • IPv6 MLD • IPv6 Neighbor Solicitation
Hardware Arbitration	Yes	Supports NC-SI hardware arbitration.

1. *TCTL.EN* should be set to 1b to activate Tx-related counters and *RCTL.RXEN*, *PRT_MNG_MANC.RCV_EN* or *WUC.APME* should be set to enable Rx-related counters.
2. As described in the Get Controller Packet Statistics Counter Numbers table in NC-SI specification.
3. The X710-TM4 does not increment the NC-SI control packets dropped counter when packets with checksum errors are dropped. In this case, only the NC-SI command checksum errors counter is updated.

9.6.2.1.1 Error Conditions

In case the device is not able to respond to a command within the 150 ms limit (three 50 ms retries), it will respond with a response code of command unavailable (0x02) and a reason code of package not ready (0x04)

9.6.2.2 AEN Handling

Asynchronous events might occur when the device is not allowed to send them via the AEN Enable command. The following rules defines the behavior of the X710-TM4 in these cases:

1. While the device is disabled, for each type of AEN only the last event is kept.
2. Outstanding AENs that occurred while a package was deselected is transmitted when a package is selected.
3. On a transition from channel disabled to channel enabled, all outstanding events are erased to prevent stale event notifications.
4. If the AEN becomes outdated before being sent (for example a link down, link up sequence occurring before the AEN is sent), then no AEN is sent.



9.6.2.2.1 Driver Status Change AEN Generation

The Host NC Driver Status changes and the Driver Status Change AEN is emitted in the following conditions:

Driver is Active — When a Driver Version AQ command is received from the PF associated with this channel.

Driver is Not Active — When a Queue Shutdown AQ command is received from the admit queue of the PF associated with this channel.

9.6.2.3 NC-SI 1.1 New Features

The following features are available when NC-SI version is set to 1.1 in the NC-SI version flag in the NVM NC-SI Configuration 1 word:

- Broadcast and multicast packets reception is enabled by default at initial state after channel is enabled.
- The Get Package UUID command and Get Package Status command are supported. The UUID reported is the same as described in [Section 9.7.8.1.2](#).
- New format for the Set Link and Get Link commands including support for new speeds.
- The following filters are added to the Enable Global Multicast Filter command:
 - DHCPv6 multicast packets from server to clients listening on well-known UDP ports
 - IPv6 MLD
 - IPv6 Neighbor Solicitation
- New format for the Capabilities Flags word in Get Capabilities response packet.

9.6.3 External Link Control Via NC-SI

9.6.3.1 NC-SI Link State Control

In NC-SI mode, the device might dynamically change the PHY power mode according to the NC-SI channel state assuming no other functionality requires the PHY to be active (host or wake up).

The following algorithm is used to define if PHY activity is required:

- At initialization time, the PHY is required to be active only if the *EMP_LINK_ON* bit in Common Firmware Parameters 2 NVM word is set.
- Once a channel is enabled via a Enable Channel NC-SI command, The PHY is powered up.
- If the channel is disabled via a Disable Channel command with the *ALD* bit set, the PHY is disabled.
- If the channel is disabled via a Reset Channel command, the PHY power state is set back to the initial value as define by the *EMP_LINK_ON* bit.

Note: Before transitioning to D3 it is the responsibility of the software device driver to request the PHY to be active for wake-up activities.



9.6.3.1.1 Get Link 2.5 Gb/s

- 0xE — 2.5 Gb/s (optional for NC-SI 1.1, reserved for NC-SI 1.0) - this new value is relevant for the X710-TM4.
- 0xF — Use values defined in *Enhanced Speed and Duplex* field starting at bit 24 (optional for NC-SI 1.1, reserved for NC-SI 1.0).
 - Set *Extended Speed and Duplex* (bits 31..24) to 0x0F.

9.6.3.1.2 Set Link 2.5/5 Gb/s Support

The Set Link command Link Setting word is modified as follows:

Bit Position	Field Description	Value Description
16:13	Additional Link Speeds (see Link Speed Selection)	Bit 15: 1b = Enable 2.5 Gb/s (optional for NC-SI 1.1, reserved for NC-SI 1.0). This a new value for the X710-TM4. Bit 16: 1b = Enable 5 Gb/s (optional for NC-SI 1.1, reserved for NC-SI 1.0). This a new value for the X710-TM4.

9.6.3.2 Set Link Error Codes

The following rules are used to define the error code returned for a Set Link command in case an invalid configuration is requested:

1. Host Driver Check — If a host device driver is present, return a Command Specific Response (0x9) with a Set Link Host OS/Driver Conflict Reason (0x1).
2. Speed Present Check — If no speed is selected, return a General Reason Code for a failed command (0x1) with Parameter Is Invalid, Unsupported, or Out-of-Range Reason (0x2).
3. Parameter Validity:
 - a. Auto-negotiation Parameter Validation — If auto-negotiation is requested and none of the selected parameters are valid for the device, return a General Reason Code for a failed command (0x1) with a Parameter Is Invalid, Unsupported, or Out-of-Range Reason (0x2).

Note: This means, for example, a command requesting 10 GbE on a 1 GbE device succeeds provided that the command requests at least one other supported speed.

Note: The setting of auto-negotiation enabled field is ignored. For speed and connection types that accept only force mode, force mode is used and for modes that supports only AN, the AN is used to enforce a speed by negotiating only this speed. There are no modes supported by the X710-TM4 that supports both AN and force modes.

The same goes for an unsupported duplex setting (a device with no HD support accepts a command with both FD and HD set), and also for HD being requested with speeds of 1 GbE and higher as long as a speed below 1 GbE is also requested (and is supported in HD). The device simply ignores the unsupported parameters.

- b. Force Mode Parameter Validation:



- If more than one link speed is being forced, then return a General Reason Code for a failed command (0x1) and a Command Specific Reason with a Set Link Speed Conflict Error (0x0905).
 - If more than one duplex setting is being forced, then return a General Reason Code for a failed command (0x1) with Parameter Is Invalid, Unsupported, or Out-of-Range Reason (0x2).
 - If 1 GbE and above is requested with HD, then return a General Reason Code for a failed command (0x1) and a Command Specific Reason with Set Link Parameter Conflict Reason (0x0903).
4. Media Type Compatibility Check — If current media type is not compatible for the requested link parameters, return a General Reason Code for a failed command (0x1) and a Command Specific Reason with Set Link Media Conflict Error (0x0902).
 5. Power State Compatibility Check — If current power state does not allow for the requested link parameters, return a General Reason Code for a failed command (0x1) and a Command Specific Reason with Set Link Power Mode Conflict Reason (0x0904).
 6. If for some reason the hardware cannot perform the flow required for the command, return a General Reason Code for a failed command (0x1) and a Command Specific Response (0x9) with Link Command Failed-Hardware Access Error (0x6).

9.6.3.3 MC External Link Control

The MC can use the NC-SI Set Link command to control the external interface link settings. This command enables the MC to set the auto-negotiation, link speed, duplex, and other parameters.

This command is only available when the host operating system is not present. Indicating the host operating system status can be obtained via the Get Link Status command and/or Host OS Status Change AEN command.

Recommendation:

- Unless explicitly needed, it is not recommended to use this feature. The NC-SI Set Link command does not expose all the possible link settings and/or features. This might cause issues under different scenarios. Even if you decided to use this feature, use it only if the link is down (trust the X710-TM4 until proven otherwise).
- It is recommended that the MC first query the link status using the Get Link Status command. The MC should then use this data as a basis and change only the needed parameters when issuing the Set Link command.

For details, refer to the NC-SI specification.

9.6.3.3.1 Set Link While LAN PCIe Functionality is Disabled

In cases where the X710-TM4 is used solely for manageability and its LAN PCIe function is disabled, using the NC-SI Set Link command while advertising multiple speeds and enabling auto-negotiation results in the lowest possible speed chosen.

To enable a higher link speed, the MC should not advertise speeds that are below the desired link speed, as the lowest advertised link speed is chosen.

When the X710-TM4 is only used for manageability and the link speed advertisement is configured by the MC, changes in the power state of the LAN device is not effected and the link speed is not re-negotiated by the LAN device.



9.6.4 NC-SI Mode – Intel-specific Commands

In addition to regular NC-SI commands, the following Intel vendor specific commands are supported. The purpose of these commands is to provide a means for the MC to access some of the Intel-specific features present in the X710-TM4.

9.6.4.1 Overview

The following features are available via the NC-SI OEM specific commands:

- Receive filters
- Packet addition decision filters 0x0...0x4
- Packet reduction decision filters 0x5...0x7
- PRT_MNG_MNGONLY register (controls the forwarding of manageability packets to the host)
- Flex 128 filters
- Flex TCP/UDP port filters 0x0...0x2
- IPv4/IPv6 filters
- Get system MAC address — This command enables the MC to retrieve the system MAC address used by the MC. This MAC address can be used for a shared MAC address mode.
- Keep PHY link up (Veto bit) enable/disable — This feature enables the MC to block PHY reset, which might cause session loss.
- TCO reset — Enables the MC to reset the X710-TM4.
- Checksum offloading — Offloads IP/UDP/TCP checksum checking from the MC.
- OS2BMC control commands.
- Firmware version commands.
- Shared MAC and shared IP commands.

These commands are designed to be compliant with their corresponding SMBus commands (if existing). All of the commands are based on a single DMTF defined NC-SI command, known as OEM Command. This command is as follows.

9.6.4.1.1 OEM Command (0x50)

The OEM command can be used by the MC to request the sideband interface to provide vendor-specific information. The Vendor Enterprise Number (VEN) is the unique MIB/SNMP private enterprise number assigned by IANA per organization. Vendors are free to define their own internal data structures in the vendor data fields.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...	Intel Command Number	Optional Data		



	Bits	
...	...	
...	Optional Data	Padding to 32 bits (0x00)
...	Checksum	

9.6.4.1.2 OEM Response (0xD0)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	Intel Command Number	Optional Return Data		
...	...			
...	Optional Return Data		Padding to 32 bits (0x00)	
...	Checksum			

Note: Responses have no command-specific reason code, unless otherwise specified within the command.

Note: The commands/responses described as follows includes only the part up to the data. The padding and checksum are implied.

9.6.4.2 OEM Commands Summary

Table 9-30. OEM Specific Command Response Reason Codes

Response Code		Reason Code	
Value	Description	Value	Description

**Table 9-30. OEM Specific Command Response Reason Codes**

Response Code		Reason Code	
0x1	Command Failed	0x5081	Invalid Intel command number.
		0x5082	Invalid Intel command parameter number.
		0x5085	Internal network controller error.
		0x5086	Invalid vendor enterprise code.
		0x508D	Returned when one of the shared IP commands is received with an out of range resource (IP, port, binding) index.
		0x508E	Returned when a request to disable a port or an IP address used in a active binding is received.
		0x5090	Returned when a binding of a non enabled resource (MAC, VLAN, IP address, port) is required.
		0x5091	Returned when the Set Port command is received with an unsupported protocol.
		0x5092	Not in shared mode. Returned when shared mode commands are used while not in shared MAC/IP mode.
		0x008E	Returned when a request to disable a VLAN or a MAC address used in a active binding is received.

Table 9-31. OEM Commands Summary

Intel Command	Parameter	Command Name	Supported in MCTP Without PT	Section
0x00	0x00	Set IP Filters Control	No	9.6.4.3
0x01	0x00	Get IP Filters Control	No	9.6.4.4
0x02	0x0F	Set Manageability Only	No	9.6.4.5.3
	0x10	Set Flexible 128 Filter Mask and Length		9.6.4.5.5
	0x11	Set Flexible 128 Filter Data		9.6.4.5.7
	0x63	Set Flex TCP/UDP Port Filters		9.6.4.5.10
	0x64	Set Flex IPv4 Address Filters		9.6.4.5.12
	0x65	Set Flex IPv6 Address Filters		9.6.4.5.14
	0x67	Set EtherType Filter		9.6.4.5.16
	0x68	Set Packet Addition Extended Filter		9.6.4.5.18
	0x69	Set Special Filter Modifiers		9.6.4.5.20
0x03	0x0F	Get Manageability Only	No	9.6.4.6.3
	0x10	Get Flexible 128 Filter Mask and Length		9.6.4.6.5
	0x11	Get Flexible 128 Filter Data		9.6.4.6.7
	0x63	Get Flex TCP/UDP Port Filters		9.6.4.6.10
	0x64	Get Flex IPv4 Address Filters		9.6.4.6.12
	0x65	Get Flex IPv6 Address Filters		9.6.4.6.14
	0x67	Get EtherType Filter		9.6.4.6.16
	0x68	Get Packet Addition Extended Filter		9.6.4.6.18
	0x69	Get Special Filter Modifiers		9.6.4.6.20



Table 9-31. OEM Commands Summary

Intel Command	Parameter	Command Name	Supported in MCTP Without PT	Section
0x04	0x10	Set Extended Unicast Packet Reduction	No	9.6.4.7.3
	0x11	Set Extended Multicast Packet Reduction		9.6.4.7.5
	0x12	Set Extended Broadcast Packet Reduction		9.6.4.7.7
0x05	0x10	Get Extended Unicast Packet Reduction	No	9.6.4.8.1
	0x11	Get Extended Multicast Packet Reduction		9.6.4.8.3
	0x12	Get Extended Broadcast Packet Reduction		9.6.4.8.5
0x06	N/A	Get System MAC Address	Yes	9.6.4.9
0x20	N/A	Set Intel Management Control	No	9.6.4.10
0x21	N/A	Get Intel Management Control	No	9.6.4.11
0x22	N/A	TCO Reset	Yes	9.6.4.12
0x23	N/A	Enable IP/UDP/TCP Checksum Offloading	No	9.6.4.13.1
0x24	N/A	Disable IP/UDP/TCP Checksum Offloading	No	9.6.4.13.3
0x25	0x0	Set IP Address	No	9.6.4.14.1
	0x1	Get IP Address		9.6.4.14.2
	0x2	Set Port		9.6.4.14.3
	0x3	Get Port		9.6.4.14.4
	0x4	Enable Unicast Infrastructure Filter		9.6.4.14.5
	0x5	Get Shared IP Capabilities Command		9.6.4.14.6
	0x6	Shared IP Enable Broadcast filtering		9.6.4.14.7
	0x7	Shared IP Enable Global Multicast filtering		9.6.4.14.8
	0x8	Get Shared IP parameters		9.6.4.14.9
	0x9	Set Binding		9.6.4.14.10
	0xA	Get Binding		9.6.4.14.11
	0xB	Set Shared Mode		9.6.4.14.12
0x40	0x01	Enable OS2BMC flow	No	9.6.4.15.1
	0x02	Enable Network to BMC flow		9.6.4.15.3
	0x03	Enable Both Network to BMC and Host to BMC flow		9.6.4.15.5
0x40	0x04	Set BMC IP address	No	9.6.4.15.7
0x41	N/A	Get OS2BMC parameters	No	9.6.4.15.9
0x48	0x1	Get Controller Information	Yes	9.6.4.16.1

Note: All the commands are supported both over RMII NC-SI and over MCTP.

9.6.4.3 Set Intel Filters Control – IP Filters Control Command (Intel Command 0x00, Filter Control Index 0x00)

This command controls different aspects of the Intel filters.



9.6.4.3.1 Set Intel Filters Control – IP Filters Control Command

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x00	0x00	IP Filters control (3-2)	
24...27	IP Filters Control (1-0)			

Where IP Filters Control has the following format.

Bit #	Name	Description	Default Value
0	IPv4/IPv6 Mode	IPv6 (0b) = There are zero IPv4 filters and four IPv6 filters. IPv4 (1b) = There are four IPv4 filters and four IPv6 filters.	1b
1...31	Reserved		

Note: This command is kept for compatibility with other projects and has no effect in X710-TM4.

9.6.4.3.2 Set Intel Filters Control – IP Filters Control Response

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x00	0x00		

9.6.4.4 Get Intel Filters Control Commands (Intel Command 0x01)

9.6.4.4.1 Get Intel Filters Control – IP Filters Control Command (Intel Command 0x01, Filter Control Index 0x00)



This command controls different aspects of the Intel filters.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x01	0x00		

9.6.4.4.2 Get Intel Filters Control – IP Filters Control Response (Intel Command 0x01, Filter Control Index 0x00)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x01	0x00	IP Filters Control (3-2)	
28...29	IP Filters Control (1-0)			

Note: This command is kept for compatibility with other projects and returns always 0x1 in X710-TM4.

9.6.4.5 Set Intel Filters Formats

9.6.4.5.1 Set Intel Filters Command (Intel Command 0x02)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x02	Parameter Number	Filters Data (optional)	



9.6.4.5.2 Set Intel Filters Response (Intel Command 0x02)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...	0x02	Filter Control Index	Return Data (Optional)	

9.6.4.5.3 Set Intel Filters – Manageability Only Command (Intel Command 0x02, Filter Parameter 0x0F)

This command sets the PRT_MNG_MNGONLY register. The PRT_MNG_MNGONLY register controls whether PT packets destined to the MC are not forwarded to the Host operating system. The PRT_MNG_MNGONLY register is listed in [Table 9-7](#).

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x02	0x0F	Manageability Only (3-2)	
24...25	Manageability Only (1-0)			

9.6.4.5.4 Set Intel Filters – Manageability Only Response (Intel Command 0x02, Filter Parameter 0x0F)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x0F		



9.6.4.5.5 Set Intel Filters – Flex Filter Enable Mask and Length Command (Intel Command 0x02, Filter Parameter 0x10)

The following command sets the Intel flex filters mask and length.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x02	0x10	Mask Byte 1	Mask Byte 2
24...27
28...31
32...35
36...37	Mask Byte 15	Mask Byte 16	Reserved	Reserved
38	Length			

9.6.4.5.6 Set Intel Filters – Flex Filter Enable Mask and Length Response (Intel Command 0x02, Filter Parameter 0x10)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x10		

9.6.4.5.7 Set Intel Filters – Flex Filter Data Command (Intel Command 0x02, Filter Parameter 0x11)

Table 9-32. Filter Data Group

Code	Bytes Programmed	Filter Data Length
0x0	bytes 0-29	1 - 30
0x1	bytes 30-59	1 - 30



Table 9-32. Filter Data Group

Code	Bytes Programmed	Filter Data Length
0x2	bytes 60-89	1 - 30
0x3	bytes 90-119	1 - 30
0x4	bytes 120-127	1 - 8

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...	0x02	0x11	Filter Data Group	Filter Data 1
	...	Filter Data N		

Note: Using this command to configure the filters data must be done after the flex filter mask command is issued and the mask is set.

9.6.4.5.8 Set Intel Filters – Flex Filter Data Response (Intel Command 0x02, Filter Parameter 0x11)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x11		

9.6.4.5.9 Set Intel Filters – Packet Addition Decision Filter Command (Intel Command 0x02, Filter Parameter 0x61)

This command is no longer supported. Use the Set Intel Filters - Packet Addition Extended Decision Filter Command (Intel Command 0x02, Filter Parameter 0x68 - [Section 9.6.4.5.18](#)) instead.



9.6.4.5.10 Set Intel Filters – Flex TCP/UDP Port Filter Command (Intel Command 0x02, Filter Parameter 0x63)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x02	0x63	Port filter index	TCP/UDP Port MSB
24..27	TCP/UDP Port LSB	Port flags		

Filter index range: 0x0...0xA. When NC-SI 1.1 is supported the range is 0x0...0x9.

Port flags are as follows:

- Bit 0: Match UDP ports
- Bit 1: Match TCP ports
- Bit 2: Match destination port (0) or source port (1).
- Bit 7:3: Reserved

If flags are not present (payload length = 9), the match is done on TCP and UDP destination ports (legacy behavior).

If the filter index is larger than 10, a command failed response code is returned with Invalid Intel Parameter Number reason (0x5082).

9.6.4.5.11 Set Intel Filters – Flex TCP/UDP Port Filter Response (Intel Command 0x02, Filter Parameter 0x63)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x63		



9.6.4.5.12 Set Intel Filters – IPv4 Filter Command (Intel Command 0x02, Filter Parameter 0x64)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x02	0x64	IP Filter Index	IPv4 Address (3)
24...26	IPv4 Address (2-0)			

Filter index range: 0x0...0x3.

9.6.4.5.13 Set Intel Filters – IPv4 Filter Response (Intel Command 0x02, Filter Parameter 0x64)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x64		

If the IP filter index is larger than 3, a command failed response code is returned Invalid Intel Parameter Number reason (0x5082).

9.6.4.5.14 Set Intel Filters – IPv6 Filter Command (Intel Command 0x02, Filter Parameter 0x65)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x02	0x65	IP filter index	...IPv6 Address (MSB, byte 15)
24...27



	Bits			
28...31
32...35
36...37	...		IPv6 Address (LSB, byte 0)	

9.6.4.5.15 Set Intel Filters – IPv6 Filter Response (Intel Command 0x02, Filter Parameter 0x65)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x65		

If the IP filter index is larger than 3, a command failed Response Code is returned, Invalid Intel Parameter Number reason (0x5082).

9.6.4.5.16 Set Intel Filters - EtherType Filter Command (Intel Command 0x02, Filter Parameter 0x67)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x02	0x67	EtherType Filter Index	EtherType Filter MSB
24...27	EtherType Filter LSB	

Where the EtherType filter has the format as described in [Section 10.2.1.21.10](#).



9.6.4.5.17 Set Intel Filters - EtherType Filter Response (Intel Command 0x02, Filter Parameter 0x67)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x67		

If the EtherType filter index is different than 2 or 3, a command failed Response Code is returned Invalid Intel Parameter Number reason (0x5082).

9.6.4.5.18 Set Intel Filters - Packet Addition Extended Decision Filter Command (Intel Command 0x02, Filter Parameter 0x68)

See [Figure 9-3](#) for description of the decision filters structure.

The command must overwrite any previously stored value. The value set is not checked.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x02	0x68	Extended Decision filter Index	Extended Decision filter 1 MSB
24...27	Extended Decision filter 1 LSB	Extended Decision filter 0 MSB
28...30	Extended Decision filter 0 LSB	

Extended decision filter index range: 0...4

Filter 0: See [Table 9-33](#).

Filter 1: See [Table 9-34](#).



Table 9-33. Filter Values

Bit #	Name	Description
3:0	Unicast (AND)	If set, packets must match unicast filter 0 to 3, respectively.
4	Broadcast (AND)	If set, packets must match the broadcast filter.
12:5	VLAN (AND)	If set, packets must match VLAN filter 0 to 7, respectively.
16:13	IPv4 Address (AND)	If set, packets must match IPv4 filter 0 to 3, respectively
20:17	IPv6 Address (AND)	If set, packets must match IPv4 filter 0 to 3, respectively
24:21	Unicast (OR)	If set, packets can pass if match unicast filter 0 to 3, respectively or a different OR filter.
25	Broadcast (OR)	If set, packets can pass if match the broadcast filter or a different OR filter.
26	Multicast (AND)	If set, packets must match the multicast filter.
27	ARP Request (OR)	If set, packets can pass if match the ARP request filter or a different OR filter.
28	ARP Response (OR)	If set, packets can pass if match the ARP response filter or a different OR filter.
29	Neighbor Discovery - 134 (OR)	If set, packets can pass if match the neighbor discovery filter(type134 - router advertisement) or a different OR filter.
30	Port 0x298 (OR)	If set, packets can pass if match a fixed TCP/UDP port 0x298 filter or a different OR filter.
31	Port 0x26F (OR)	If set, packets can pass if match a fixed TCP/UDP port 0x26F filter or a different OR filter.

Table 9-34. Extended Filter 1 Values

Bit #	Name	Description
3:0	Ethertype 0 -3 (AND)	If set, packets must match the Ethertype filter 0 to 3, respectively.
7:4	Ethertype 0 -3 (OR)	If set, packets must match the Ethertype filter 0 to 3, respectively or a different OR filter.
18:8	Flex port 10:0 (OR)	If set, packets can pass if match the TCP/UDP Port filter 10:0.
19	DHCPv6 (OR)	If set, packets can pass if match the DHCPv6 port (0x0223).
20	DHCP Client (OR)	If set, packets can pass if match the DHCP Server port (0x0043).
21	DHCP Server (OR)	If set, packets can pass if match the DHCP Client port (0x0044).
22	NetBIOS Name Service (OR)	If set, packets can pass if match the NetBIOS Name Service port (0x0089).
23	NetBIOS Datagram Service (OR)	If set, packets can pass if match the NetBIOS Datagram Service port (0x008A).
24	Flex TCO (OR)	If set, packets can pass if match the Flex 128 TCO filter.
25	Neighbor Discovery - 135 (OR)	If set, packets must also match the neighbor discovery filter (type135 - Neighbor Solicitation). or a different OR filter.
26	Neighbor Discovery - 136 (OR)	If set, packets must also match the neighbor discovery filter (type136 - Neighbor Advertisement) or a different OR filter.
27	Neighbor Discovery - 137 (OR)	If set, packets must also match the neighbor discovery filter (type137 - Redirect) or a different OR filter.
28	ICMPv4 (OR)	Controls the inclusion of ICMPv4 filtering in the manageability filter decision (OR section).
29	MLD	If set, packets must also match one of the MLD ICMPv6 types or a different OR filter.
31:30	Reserved	Reserved



9.6.4.5.19 Set Intel Filters – Packet Addition Extended Decision Filter Response (Intel Command 0x02, Filter Parameter 0x68)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x68		

If the extended decision filter index is larger than 5, a command failed Response Code is returned Invalid Intel Parameter Number reason (0x5082).

9.6.4.5.20 Set Intel Filters - Special Modifier Command (Intel Command 0x02, Filter Parameter 0x69)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x02	0x69	Special Modifier Register MSB	
24...27	Special Modifier Register LSB		Padding	

Where the special modifier filter has the format as described in [Section 10.2.1.21.14](#). The value set is not checked.

9.6.4.5.21 Set Intel Filters - Special Modifier Response (Intel Command 0x02, Filter Parameter 0x69)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x02	0x69		



9.6.4.6 Get Intel Filters Formats

9.6.4.6.1 Get Intel Filters Command (Intel Command 0x03)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x03	Parameter Number		

9.6.4.6.2 Get Intel Filters Response (Intel Command 0x03)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x03	Parameter Number	Optional Return Data	

9.6.4.6.3 Get Intel Filters – Manageability Only Command (Intel Command 0x03, Filter Parameter 0x0F)

This command retrieves the PRT_MNG_MNGONLY register. The PRT_MNG_MNGONLY register controls whether PT packets destined to the MC are also be forwarded to the host operating system.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x03	0x0F		

9.6.4.6.4 Get Intel Filters – Manageability Only Response (Intel Command 0x03, Filter Parameter 0x0F)

The PRT_MNG_MNGONLY register structure is listed in [Table 9-7](#).



	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x03	0x0F	Manageability to Host (3-2)	
28...29	Manageability to Host (1-0)			

9.6.4.6.5 Get Intel Filters – Flex Filter 0 Enable Mask and Length Command (Intel Command 0x03, Filter Parameter 0x10)

The following command retrieves the Intel flex filters mask and length. See [Section 9.3.3.6](#) for details of the values returned by this command.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x03	0x10		

9.6.4.6.6 Get Intel Filters – Flex Filter 0 Enable Mask and Length Response (Intel Command 0x03, Filter Parameter 0x10)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x03	0x10	Mask Byte 1	Mask Byte 2
28...31
32...35
36...39
40...43	...	Mask Byte 16	Reserved	Reserved
44	Flexible Filter Length			



9.6.4.6.7 Get Intel Filters – Flex Filter 0 Data Command (Intel Command 0x03, Filter Parameter 0x11)

The following command retrieves the Intel flex filters data.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x03	0x11	Filter Data Group 0...4	

The filter data group parameter defines which bytes of the flex filter are returned by this command:

Table 9-35. Filter Data Group

Code	Bytes Returned
0x0	bytes 0-29
0x1	bytes 30-59
0x2	bytes 60-89
0x3	bytes 90-119
0x4	bytes 120-127

9.6.4.6.8 Get Intel Filters – Flex Filter 0 Data Response (Intel Command 0x03, Filter Parameter 0x11)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...	0x03	0x11	Filter Group Number	Filter Data 1
	...	Filter Data N		

9.6.4.6.9 Get Intel Filters – Packet Addition Decision Filter Command (Intel Command 0x03, Filter Parameter 0x61)

This command is no longer supported. Use the Get Intel Filters - Packet Addition Extended Decision Filter Command (Intel Command 0x03, Filter Parameter 0x68 - [Section 9.6.4.6.18](#)) instead.



9.6.4.6.10 Get Intel Filters – Flex TCP/UDP Port Filter Command (Intel Command 0x03, Filter Parameter 0x63)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...22	0x03	0x63	TCP/UDP Filter Index	

Filter index range: 0x0...0x2.

9.6.4.6.11 Get Intel Filters – Flex TCP/UDP Port Filter Response (Intel Command 0x03, Filter Parameter 0x63)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x03	0x63	TCP/UDP Filter Index	TCP/UDP Port (1)
28..29	TCP/UDP Port (0)	Port flags		

Filter index range: 0x0...0x2.

9.6.4.6.12 Get Intel Filters – IPv4 Filter Command (Intel Command 0x03, Filter Parameter 0x64)

	Bits			
Bytes	31...24	23...16	15...08	07...00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...22	0x03	0x64	IPv4 Filter Index	

Note: The filters index range can vary according to the IPv4/IPv6 mode setting in the Filters Control command.

IPv4 Mode: Filter index range: 0x0...0x3.

IPv6 Mode: This command should not be used in IPv6 mode.



9.6.4.6.13 Get Intel Filters – IPv4 Filter Response (Intel Command 0x03, Filter Parameter 0x64)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x03	0x64	IPv4 Filter Index	IPv4 Address (3)
28...29	IPv4 Address (2-0)			

9.6.4.6.14 Get Intel Filters – IPv6 Filter Command (Intel Command 0x03, Filter Parameter 0x65)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...22	0x03	0x65	IPv6 Filter Index	

Note: The filters index range can vary according to the IPv4/IPv6 mode setting in the Filters Control command

IPv4 Mode: Filter index range: 0x0...0x2.

IPv6 Mode: Filter index range: 0x0...0x3.

9.6.4.6.15 Get Intel Filters – IPv6 Filter Response Intel Command 0x03, Filter Parameter 0x65)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x03	0x65	IPv6 Filter Index	IPv6 Address (MSB, Byte 16)
28...31



	Bits			
Bytes	31:24	23:16	15:08	07:00
32...35
36...39
40...42	IPv6 Address (LSB, Byte 0)	

9.6.4.6.16 Get Intel Filters - EtherType Filter Command (Intel Command 0x03, Filter Parameter 0x67)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...22	0x03	0x67	EtherType Filter Index	

Valid indices: 0...3

9.6.4.6.17 Get Intel Filters - EtherType Filter Response (Intel Command 0x03, Filter Parameter 0x67)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x03	0x67	EtherType Filter Index	EtherType Filter MSB
28...30	EtherType Filter LSB	

If the Ethertype filter index is larger than 3, a command failed Response Code is returned Invalid Intel Parameter Number reason (0x5082).

9.6.4.6.18 Get Intel Filters – Packet Addition Extended Decision Filter Command (Intel Command 0x03, Filter Parameter 0x68)

This command enables the MC to retrieve the extended decision filter.



	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...22	0x03	0x68	Extended Decision Filter Index	

9.6.4.6.19 Get Intel Filters – Packet Addition Extended Decision Filter Response (Intel Command 0x03, Filter Parameter 0x68)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x03	0x68	Decision Filter Index	Decision Filter 1 MSB
28...31	Decision Filter 1 LSB	Decision Filter 0 MSB
32...34	Decision Filter 0 LSB	

Where decision filter 0 and decision filter 1 have the structure as detailed in the respective Set commands.

If the extended decision filter index is larger than 4, a command failed Response Code is returned Invalid Intel Parameter Number reason (0x5082).

9.6.4.6.20 Get Intel Filters – Special Modifier Command (Intel Command 0x03, Filter Parameter 0x69)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x03	0x69	Padding	

Where the special modifier filter has the format as described in [Section 10.2.1.21.14](#).



9.6.4.6.21 Get Intel Filters - Special Modifier Response (Intel Command 0x02, Filter Parameter 0x69)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x03	0x69	Special Modifier Register MSB	
28...29	Special Modifier Register LSB		Padding	

9.6.4.7 Set Intel Packet Reduction Filters Formats

The non-extended commands are obsolete. The extended commands (Section 9.6.4.7.3 to Section 9.6.4.7.8) should be used instead.

9.6.4.7.1 Set Intel Packet Reduction Filters Command (Intel Command 0x04)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x04	Packet Reduction Index	Packet Reduction Data...	

Note: It is advised that the MC only use the extended packet reduction commands.

The *Packet Reduction* data field has the following structure:

Table 9-36. Packet Reduction Field Description

Bit #	Name	Description
12:0	Reserved	Reserved
16:13	IPv4 Address (AND)	If set, packets must match IPv4 filter 0 to 3, respectively.
20:17	IPv6 Address (AND)	If set, packets must match IPv4 filter 0 to 3, respectively.
27:21	Reserved	Reserved.
28	ARP Response (OR)	If set, packets can pass if match the ARP response filter or a different OR filter.



Table 9-36. Packet Reduction Field Description

Bit #	Name	Description
29	Reserved	Reserved.
30	Port 0x298	If set, packets can pass if match a fixed TCP/UDP port 0x298 filter.
31	Port 0x26F	If set, packets can pass if match a fixed TCP/UDP port 0x26F filter.

Table 9-37. Extended Packet Reduction Field Description

Bit #	Name	Description
3:0	Ethertype 0 -3 (AND)	If set, packets must match the Ethertype filter 0 to 3, respectively.
7:4	Ethertype 0-3 (OR)	If set, packets can pass if match the Ethertype filter 0 to 3, respectively.
15:12	Reserved	Reserved.
8:18	Flex port 10:0 (OR)	If set, packets can pass if match the TCP/UDP Port filter 10:0.
23:19	Reserved	Reserved.
24	Flex TCO (OR)	If set, packets can pass if match the Flex 128 TCO filter.
27:25	Reserved	Reserved.
28	ICMPv4	Is set, ICMPv4 packets can pass.
31:29	Reserved	Reserved.

The filtering is divided into two decisions:

- Bit 20:13 in Table 9-36 and Bits 3:2 in Table 9-37 works in an AND manner; it must be true in order for a packet to pass (if was set).

Bits 28 in Table 9-36 and Bits 24:10 in Table 9-37 work in an OR manner; at least one of them must be true for a packet to pass (if any were set).

9.6.4.7.2 Set Intel Packet Reduction Filters Response (Intel Command 0x04)

Bytes	Bits			
	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...	0x04	Packet Reduction Index		

9.6.4.7.3 Set Unicast Extended Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x10)

The command has the following format:



	Bits			
Bytes	31:24	23:16	15:08	07:00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x04	0x10	Extended Unicast Reduction Filter MSB	..
24..27	..	Extended Unicast Reduction Filter LSB	Unicast Reduction Filter MSB	..
28..29	..	Unicast Reduction Filter LSB		

The command overwrites any previously stored value.

Note: See [Table 9-36](#) and [Table 9-37](#) for description of the unicast extended packet reduction format.

9.6.4.7.4 Set Unicast Extended Packet Reduction Response (Intel Command 0x04, Reduction Filter Index 0x10)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x04	0x10		



9.6.4.7.5 Set Multicast Extended Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x11)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x04	0x11	Extended Multicast Reduction Filter MSB	..
24..27	..	Extended Multicast Reduction Filter LSB	Multicast Reduction Filter MSB	..
28..29	..	Multicast Reduction Filter LSB		

Note: See [Table 9-36](#) and [Table 9-37](#) for description of the multicast extended packet reduction format.

The command overwrites any previously stored value.

9.6.4.7.6 Set Multicast Extended Packet Reduction Response (Intel Command 0x04, Reduction Filter Index 0x11)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x04	0x11		



9.6.4.7.7 Set Broadcast Extended Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x12)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x04	0x12	Extended Broadcast Reduction Filter MSB	..
24..27	..	Extended Broadcast Reduction Filter LSB	Broadcast Reduction Filter MSB	..
28..29	..	Broadcast Reduction Filter LSB		

Note: See [Table 9-36](#) and [Table 9-37](#) for description of the broadcast extended packet reduction format.

The command overwrites any previously stored value.

9.6.4.7.8 Set Broadcast Extended Packet Reduction Response (Intel Command 0x04, Reduction Filter Index 0x12)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x04	0x12		

9.6.4.8 Get Intel Packet Reduction Filters Formats

Note: The non extended commands are obsolete. Use the extended commands ([Section 9.6.4.8.1](#) to [Section 9.6.4.8.6](#)) instead.



9.6.4.8.1 Get Unicast Extended Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x10)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x05	0x10		

9.6.4.8.2 Get Unicast Extended Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x10)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x05	0x00	Extended Unicast Packet Reduction (3-2)	
28...29	Extended Unicast Packet Reduction (1-0)		Unicast Packet Reduction (3-2)	
30...31	Unicast Packet Reduction (1-0)			

9.6.4.8.3 Get Multicast Extended Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x11)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x05	0x11		



9.6.4.8.4 Get Multicast Extended Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x11)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x05	0x11	Extended Multicast Packet Reduction (3-2)	
28...29	Extended Multicast Packet Reduction (1-0)		Multicast Packet Reduction (3-2)	
30...31	Multicast Packet Reduction (1-0)			

9.6.4.8.5 Get Broadcast Extended Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x12)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x05	0x12		

9.6.4.8.6 Get Broadcast Extended Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x12)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x05	0x12	Extended Broadcast Packet Reduction (3-2)	
28...29	Extended Broadcast Packet Reduction (1-0)		Broadcast Packet Reduction (3-2)	
30...31	Broadcast Packet Reduction (1-0)			



9.6.4.9 System MAC Address

9.6.4.9.1 Get System MAC Address Command (Intel Command 0x06)

In order to support a system configuration that requires the NC to hold the MAC address for the MC (such as shared MAC address mode), the following command is provided to enable the MC to query the NC for a valid MAC address.

The NC must return the system MAC addresses. The MC should use the returned MAC addressing as a shared MAC address by setting it using the Set MAC Address command as defined in NC-SI 1.0.

When a single function is defined on the port, it returns the LAN MAC address of this function as read from the PF allocations NVM section or from the alternate RAM or as set by the Manage MAC address write AQ command. When more than one function is defined on the port, it returns the address of the lowest defined function on this port.

It is also recommended that the MC use packet reduction and the Manageability-to-Host command to set the proper filtering method.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20	0x06			

9.6.4.9.2 Get System MAC Address Response (Intel Command 0x06)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x06	MAC Address		
28...30	MAC Address			



9.6.4.10 Set Intel Management Control Formats

9.6.4.10.1 Set Intel Management Control Command (Intel Command 0x20)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...22	0x20	0x00	Intel Management Control 1	

Where Intel Management Control 1 is as follows:

Bit #	Default value	Description
0	0b	Enable Critical Session Mode (Keep PHY Link Up and Veto Bit). 0b = Disabled. 1b = Enabled. When critical session mode is enabled, the following behaviors are disabled: <ul style="list-style-type: none"> The PHY is not reset on PE_RST_N and PCIe resets (in-band and link drop). Other reset events are not affected – Internal_Power_On_Reset, device disable, Force TCO, and PHY reset by software. The PHY does not change its power state. As a result, link speed does not change. The device does not initiate configuration of the PHY to avoid losing link.
7:1	0x0	Reserved.

9.6.4.10.2 Set Intel Management Control Response (Intel Command 0x20)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x20	0x00		



9.6.4.11 Get Intel Management Control Formats

9.6.4.11.1 Get Intel Management Control Command (Intel Command 0x21)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x21	0x00		

Where Intel Management Control 1 is as described in [Section 9.6.4.10.2](#).

9.6.4.11.2 Get Intel Management Control Response (Intel Command 0x21)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...26	0x21	0x00	Intel Management Control 1	

9.6.4.12 TCO Reset

Depending on the bit set in the TCO mode field this command causes the X710-TM4 to perform either:

1. TCO reset

- If force TCO reset is enabled in the NVM (see [Section 6.2.36.2](#)). The force TCO reset clears the data path (Rx/Tx) of the X710-TM4 to enable the MC to transmit/receive packets through the X710-TM4.
- If the MC has detected that the operating system is hung and has blocked the Rx/Tx path, the force TCO reset clears the data-path (Rx/Tx) of the NC to enable the MC to transmit/receive packets through the NC.
- After successfully performing the command, the NC considers the Force TCO command as an indication that the operating system is hung and clears the internal driver up indication. If TCO reset is disabled in the NVM, the X710-TM4 does not reset the data path and notifies the MC on successful completion.



2. TCO isolate

- If TCO isolate is enabled in the NVM (see [Section 6.2.36.3](#)). The TCO Isolate command disables PCIe write operations to the LAN port.
- If TCO isolate is disabled in NVM, the X710-TM4 does not execute the command but sends a response to the MC with successful completion.
- Once TCO isolate is set, the driver is supposed to be disabled, and it is reported as such to the MC.

3. Firmware reset

- This command causes re-initialization of all the manageability functions and re-loads of manageability related NVM words (such as firmware patch code).
- When the MC loads a new management related NVM image (like a firmware patch) the Firmware Reset command loads the management related NVM information without the need to power down the system.
- This command is issued to the package and affects all channels. After the firmware reset, the firmware Semaphore register (FWSM) is re-initialized.
- Applying this command resets the entire device and also has an effect on TCO reset.

Note: TCO isolate affects only the channel (port) that the command was issued to. Force TCO resets the entire device (all channels in the package).

Following firmware reset, the MC needs to re-initialize all ports. A firmware reset causes a global reset of the entire device (GLOBR).

Note: Only one of the fields should be set in a given command. Setting more than one field might yield unexpected results.

9.6.4.12.1 Perform Intel TCO Reset Command (Intel Command 0x22)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20	0x22	TCO Mode		

Where TCO mode is:



Field	Bit(s)	Description
DO_TCO_RST	0	Do TCO Reset. 0b = Do nothing. 1b = Perform TCO reset.
DO_TCO_ISOLATE ¹	1	Do TCO Isolate. 0b = Enable PCIe write access to LAN port. 1b = Isolate Host PCIe write operation to the port Note: Should be used for debug only. Note: The TCO Isolate do not impact MCTP traffic Note: When isolate is set, the OS2BMC flow is also disabled.
RESET_MGMT	2	Reset Manageability; Re-load Manageability NVM Words. 0b = Do nothing. 1b = Issue firmware reset to manageability. Setting this bit generates a one-time firmware reset. Following the reset, management related data from the NVM is loaded. Note: A reset of the internal firmware causes a reset of the entire device.
Reserved	7:3	Reserved (set to 0x00).

Note: For compatibility, the TCO Reset command without the TCO mode parameter is accepted (TCO reset is done).

1. TCO isolate host write operation enabled in the NVM.

9.6.4.12.2 Perform Intel TCO Reset Response (Intel Command 0x22)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...26	0x22			

9.6.4.13 Checksum Offloading

This command enables the checksum offloading filters in the NC.

When enabled, these filters block any packets that did not pass IP, UDP or TCP checksum from being forwarded to the MC.



9.6.4.13.1 Enable Checksum Offloading Command (Intel Command 0x23)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20	0x23			

9.6.4.13.2 Enable Checksum Offloading Response (Intel Command 0x23)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...26	0x23			

9.6.4.13.3 Disable Checksum Offloading Command (Intel Command 0x24)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20	0x24			



9.6.4.13.4 Disable Checksum Offloading Response (Intel Command 0x24)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...26	0x24			

9.6.4.14 Shared MAC and Shared IP Support Commands (Intel Command 0x25)

To meet the requirements introduced by sharing IP addresses, modifications and additions to the NC-SI command set are required. These changes include the new commands in this section and the modifications described in [Section 9.3.6](#).

Note: All indexes in this command set starts at one to match the NC-SI methodology.

9.6.4.14.1 Set IP Address Command (Intel Command 0x25, Index = 0x0)

The Set IP Address command is used by the MC to communicate its IP address to a NC. The format of a Set IP Address command packet is listed in [Table 9-38](#).

If at least one IP address filter is enabled, only unicast packets that match one of the enabled filters are forwarded through the NC-SI interface. Otherwise, the IP address is ignored in the unicast filtering process.

This command does not impact the forwarding results. It is used as a preliminary stage to the Set Binding command.

Table 9-38. Set IP Address Command Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0x0	Reserved	

**Table 9-38. Set IP Address Command Packet Format**

Bits			
24..27	Management Controller IP Address		
28..31			
32..35			
36..39			
40..43	Reserved	IP Address Number	Set IP Flags
44..47	Checksum		

- MC IP Address — An IP address that is used by the MC. If the *IP Version* bit of the *Flags* field is 0b (IPv4), this is a 4-byte unicast IPv4 address in network byte order. In this case, the address occupies bytes 24-27 of the packet, and bytes 28-39 are ignored. If the *IP Version* bit of the *Flags* field is 1b (IPv6), this is a 16-byte unicast IPv6 address in network byte order. In this case, the address occupies the full field (bytes 24-39 of the packet).
- IP Address Number — Indicates which IP address filter is configured by the command. The value can relate to one of three pools of filters according to the following table:

Table 9-39. IP Filters Pools

Set IP Flag.IP Version	Set IP Flag.Mixed Index	Pool to Use	Allowed Values
0	0	IPv4	1 to the number of IPv4 only addresses.
1	0	IPv6	1 to the number of IPv6 only addresses.
X (0/1)	1	Mixed	1 to the number of mixed IP addresses.

Note: The values shown in the allowed values column refers to the Get Shared IP Capabilities Response ([Section 9.6.4.14.6.1](#)).

- [Table 9-40](#) lists the bits fields in the *Set IP Flags* field.

Table 9-40. Set IP Flag Field

Bit Position	Field Description	Value Description
0	Enable	0b = Disable the filter. 1b = Enable the filter.
1	IP version	0b = IPv4. 1b = IPv6.
2	Mixed index	0b = Index relates to the IPv4 or IPv6 only IP filter sets according to the IP version field. 1b = Index relates to the mixed IP filter set.
3	MAC based IP	This flags define if the Ipv6 address is derived from a MAC address and thus only the 24 LSB should be used for the comparison. This flag is relevant only if the IP version = IPv6. 0b = Filter according to the full 128 bits of IPv6 address. 1b = Filter according to the 24 LS bits of the IPv6 address.
7:4	Reserved	Reserved.



9.6.4.14.1.1 Set IP Address Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Set IP Address command and send a response using the format listed in [Table 9-41](#).

Table 9-41. Set IP Address Response Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x0	Reserved	
28..31	Checksum			

9.6.4.14.2 Get IP Address Command (Intel Command 0x25, Index = 0x1)

An MC uses the Get IP Address command to determine the IP address programmed in one of the IP address filters in a NC. The format of a Get IP Address command packet is listed in [Table 9-42](#).

Table 9-42. Get IP Address Command Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0x1	Reserved	
24..27	Reserved		IP Address Number	IP filter pool
28..31	Checksum			

- IP address number. Defines the index of the IP address in the pool defined by the IP filter pool. The allowed values are listed in [Table 9-39](#).
- IP filter pool:
 - 0x0: Mixed IP filters
 - 0x1: IPv4 filters
 - 0x2: IPv6 filters
 - 0x3 - 0xFF: Reserved

9.6.4.14.2.1 Get IP Address Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Get IP Address command and send a response using the format listed in [Table 9-43](#).



Table 9-43. Get IP Address Response Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x1	IP Address Number	Get IP Flags
28..31	Management Controller IP Address			
32..35				
36..39				
40..43				
44..47				

- MC IP Address — An IP address that is used by the MC. If the *IP Version* bit of the *Flags* field is 0b (IPv4), this is a 4-byte unicast IPv4 address in network byte order. In this case, the address occupies bytes 28-31 of the packet, and bytes 32-43 are ignored. If the *IP Version* bit of the *Flags* field is 1b (IPv6), this is a 16-byte unicast IPv6 address in network byte order. In this case, the address occupies the full field (bytes 28-43 of the packet).
- IP Address Number — Indicates which IP address filter is described in the response. Should be equal to the IP address number in the command.
- Table 9-44 lists the bits fields in the *Get IP Flags* field.

Table 9-44. Get IP Flag Field

Bit Position	Field Description	Value Description
0	Enable	0b = Filter is disabled. 1b = Filter is enabled.
1	IP version	0b = IPv4. 1b = IPv6.
2	Mixed Index	0b = Index relates to the IPv4 or IPv6 only IP filter sets according to the IP version field. 1b = Index relates to the mixed IP filter set.
3	MAC based IP	This flag defines if the IPpv6 address is derived from a MAC address and thus only the 24 LSB should be used for the comparison. This flag is relevant only if the IP version = IPv6. 0b = Filter according to the full 128 bits of IPv6 address. 1b = Filter according to the 24 LS bits of the IPv6 address.
7:4	Reserved	Reserved.

9.6.4.14.3 Set Port Command (Intel Command 0x25, Index = 0x2)

An MC uses the Set Port command to communicate one of its TCP or UDP ports to a NC. The format of a Set Port command packet is listed in Table 9.6.4.14.3.1.

This command does not impact the forwarding results. It is used as a preliminary stage to the Set Binding command.



If the *Ignore Protocol* flag is cleared, the protocol should also match the *Protocol* field; otherwise, the *Protocol* field is ignored.

9.6.4.14.3.1 Set Port Command Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0x2	Set Port Flags	Reserved
24..27	Port Index	Protocol	Port	
28..31	Checksum			

- Protocol — The value to match in the IPv4 header *Protocol* field or IPv6 header *Next Header* field. These values are defined by IANA. Allowed values are 0x6 (TCP) and 0x11 (UDP).
- Port — The value to match in the *Destination Port* or *Source Port* field of the TCP or UDP header. The legal port range for both TCP and UDP is 0-65,535. The compared field is defined by the Port Type flag.
- Port Index — Indicates which port filter is configured by the command. Allowed values are 1 to *n*, where *n* is the number of port filters supported by the Network Controller.

Table 9.6.4.14.3.2 lists the fields in the *Set Port Flags* field.

9.6.4.14.3.2 Set Port Flags Field Descriptions

Bit Position	Field Description	Value Description
0	Enable	0b = Disable the filter. 1b = Enable the filter.
1	Ignore Protocol	0b = Filter by port and protocol. 1b = Filter by port only.
2	Port Type	0b = Compare destination port. 1b = Compare source port.
7:3	Reserved	Reserved.

9.6.4.14.3.3 Set Port Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Set Port command and send a response using the format listed in Table 9-45.

Table 9-45. Set Port Response Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
12..15	NC-SI Header			
16..19	Response Code		Reason Code	

**Table 9-45. Set Port Response Packet Format**

	Bits		
20..23	Manufacturer ID (Intel 0x157)		
24..27	0x25	0x2	Reserved
28..31	Checksum		

9.6.4.14.4 Get Port Command (Intel Command 0x25, Index = 0x3)

An MC uses the Get Port command to determine the TCP or UDP port programmed in one of the port filters in a NC. The format of a Get Port command packet is listed in [Table 9-46](#).

Table 9-46. Get Port Command Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0x3	Reserved	
24..27	Reserved		Port Index	Reserved
28..31	Checksum			

[Table 9-47](#) lists the fields in the Get Port command.

Table 9-47. Get Port Command Field Descriptions

Field	Field Description	Value Description
Port Index	Indicates which port filter is requested by the command.	1 to n , where n is the number of port filters supported by the NC.



9.6.4.14.4.1 Get Port Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Get Port command and send a response using the format listed in [Table 9-48](#).

Table 9-48. Get Port Response Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x3	Get Port Flags	Reserved
28..31	Port Index	Protocol	Port	
32..35	Checksum			

- Protocol — The value compared in the IPv4 header *Protocol* field or IPv6 header *Next Header* field. Possible values are 0x6 (TCP) and 0x11 (UDP).
- Port — The value compared in the *Destination Port or Source Port* field of the TCP or UDP header.
- Port Index — Indicates which port filter is reported by the response. Should match the Port Index in the command.

[Table 9-49](#) lists the fields in the *Get Port Flags* field.

Table 9-49. Get Port Flags Field Descriptions

Bit Position	Field Description	Value Description
0	Enable	0b = Filter is disabled. 1b = Filter is enabled.
1	Ignore Protocol	0b = Filter by port and protocol. 1b = Filter by port only.
2	Port Type	0b = Compare destination port. 1b = Compare source port.
7:3	Reserved	Reserved.

9.6.4.14.5 Enable Unicast Infrastructure Filter Command (Intel Command 0x25, index = 0x4)

A MC uses the Enable Unicast Infrastructure Filter command to configure a NC to forward copies of network infrastructure packets to it. Network infrastructure packets contain messages that are necessary for operating the network infrastructure layers (such as DHCP, ARP, and DNS messages). This is required when the MC shares an IP address with the host. In this case, both the host and the MC need to process the messages. As a result, the NC must forward the packets to both the MC and the host.

This command should be applied only after a MAC address is added using the Set MAC Address NC-SI command.



All the IP addresses added through the Set IP command before this command is given are considered as IP addresses of the MC for the purpose of this command.

If a Set IP command is received after this command was received, the list of IP address is not updated and this command should be given again.

The format of an Enable Unicast Infrastructure Filter command packet is listed in [Table 9-50](#).

Table 9-50. Enable Unicast Infrastructure Filter Command

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x4	Reserved	
28..31	Unicast Infrastructure Filter Settings			
32..35	Checksum			
36..29	Padding			

[Table 9-51](#) lists the sub fields of the *Unicast Infrastructure Filter Settings* field.

Table 9-51. Unicast Infrastructure Packet Filter Settings Field

Bit Position	Field Description	Value Description
0	ARP Response Packets Received From Wire	0x1 = Forward this packet type to both the host and the MC. 0x0 = Forward this packet type to the host only. For the purposes of this filter, an ARP response packet is defined to be any packet that meets all of the following requirements: <ul style="list-style-type: none"> The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. The Ethernet <i>Type</i> field contains 0x0806 (ARP). The ARP <i>Opcode</i> field is set to 0x0002 (response). The ARP <i>Target Protocol Address</i> field contains the IP address assigned to the MC.
1	ICMPv4 Request Packets Received From Wire	0x1 = Forward this packet type to both the host and the MC. 0x0 = Forward this packet type to the host only. For the purposes of this filter, an ICMP request packet is defined to be any packet that meets all of the following requirements: <ul style="list-style-type: none"> The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. The Ethernet <i>Type</i> field contains 0x0800 (IPv4). The IP <i>Destination Address</i> field contains the IPv4 address assigned to the MC. The IP <i>Protocol</i> field contains 1 (ICMP).
2	ICMPv6 Request Packets Received From Wire	0x1 = Forward this packet type to both the host and the MC. 0x0 = Forward this packet type to the host only. For the purposes of this filter, an ICMPv6 request packet is defined to be any packet that meets all of the following requirements: <ul style="list-style-type: none"> The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. The Ethernet <i>Type</i> field contains 0x86DD (IPv6). The IP <i>Destination Address</i> field contains the IPv6 address assigned to the MC. The IP <i>Next Header</i> field contains 58 (ICMPv6). Note: This filter is not supported by the X710-TM4.



Table 9-51. Unicast Infrastructure Packet Filter Settings Field

Bit Position	Field Description	Value Description
3	DHCP Server Unicast Packets Received From Wire	<p>0x1 = Forward this packet type to both the host and the MC. 0x0 = Forward this packet type to the host only.</p> <p>For the purposes of this filter, a DHCP server unicast packet is defined to be any packet that meets all of the following requirements:</p> <ul style="list-style-type: none"> • The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. • The Ethernet <i>Type</i> field contains 0x0800 (IPv4). • The IP <i>Destination Address</i> field contains either 255.255.255.255 (the local broadcast address) or the IPv4 address assigned to the MC. • The IP <i>Protocol</i> field contains 17 (UDP). • The UDP <i>Destination Port</i> field contains 68 (bootstrap protocol client).
4	DNS Server Packets Received From Wire	<p>0x1 = Forward this packet type to both the host and the MC. 0x0 = Forward this packet type to the host only.</p> <p>For the purposes of this filter, a DNS server unicast packet is defined to be any packet that meets all of the following requirements:</p> <ul style="list-style-type: none"> • The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. • The Ethernet <i>Type</i> field contains 0x0800 (IPv4). • The IP <i>Destination Address</i> field contains the IPv4 address assigned to the MC. • The IP <i>Protocol</i> field contains 17 (UDP). • The UDP <i>Source Port</i> field contains 53 (domain name server).
5	DHCP Client Packets Transmitted By Host	<p>0x1 = Forward this packet type to both the wire and the MC. 0x0 = Forward this packet type to the wire only.</p> <p>For the purposes of this filter, a DHCP client unicast packet is defined to be any packet that meets all of the following requirements:</p> <ul style="list-style-type: none"> • The Ethernet <i>Source Address</i> field contains the MAC address assigned to the MC. • The Ethernet <i>Type</i> field contains 0x0800 (IPv4). • The IP <i>Protocol</i> field contains 17 (UDP). • The UDP <i>Destination Port</i> field contains 67 (bootstrap protocol server).
6	DHCPv6 Server Unicast Packets Received From Wire	<p>0x1 = Forward this packet type to both the host and MC. 0x0 = Forward this packet type to the host only.</p> <p>For the purposes of this filter, a DHCPv6 server unicast packet is defined to be any packet that meets all of the following requirements:</p> <ul style="list-style-type: none"> • The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. • The Ethernet <i>Type</i> field contains 0x86DD (IPv6). • The IPv6 <i>Destination Address</i> field contains the IPv6 address assigned to the MC. • The IP <i>Protocol</i> field contains 17 (UDP). • The UDP <i>Destination Port</i> field contains 546 (DHCPv6 protocol client).
7	RMCP Primary Port - UDP	<p>0x1 = Forward this packet type to the MC only. 0x0 = Forward this packet type to the host.</p> <p>For the purposes of this filter, a RMCP primary UDP packet is defined to be any packet that meets all of the following requirements:</p> <ul style="list-style-type: none"> • The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. • The Ethernet <i>Type</i> field contains 0x86DD (IPv6) Or 0x0800 (IPv4). • The IP <i>Destination Address</i> field contains the one of the IP address assigned to the MC. • The IP <i>Protocol</i> field contains 17 (UDP). • The UDP <i>Destination Port</i> field contains 623 [aux bus shunt (primary RMCP port)].
8	RMCP Primary Port - TCP	<p>0x1 = Forward this packet type to the MC only. 0x0 = Forward this packet type to the host.</p> <p>For the purposes of this filter, a RMCP primary TCP packet is defined to be any packet that meets all of the following requirements:</p> <ul style="list-style-type: none"> • The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. • The Ethernet <i>Type</i> field contains 0x86DD (IPv6) Or 0x0800 (IPv4). • The IP <i>Destination Address</i> field contains the one of the IP address assigned to the MC. • The IP <i>Protocol</i> field contains 6 (TCP). • The UDP <i>Destination Port</i> field contains 623 [aux bus shunt (primary RMCP port)].

**Table 9-51. Unicast Infrastructure Packet Filter Settings Field**

Bit Position	Field Description	Value Description
9	RMCP Secondary Port - UDP	0x1 = Forward this packet type to the MC only. 0x0 = Forward this packet type to the host For the purposes of this filter, a RMCP secondary UDP packet is defined to be any packet that meets all of the following requirements: <ul style="list-style-type: none"> The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. The Ethernet <i>Type</i> field contains 0x86DD (IPv6) or 0x0800 (IPv4). The IP <i>Destination Address</i> field contains the one of the IP address assigned to the MC. The IP <i>Protocol</i> field contains 17 (UDP). The UDP <i>Destination Port</i> field contains 664 [secure aux bus (secondary RMCP port)].
10	RMCP Secondary Port - TCP	0x1 = Forward this packet type to the MC only. 0x0 = Forward this packet type to the host. For the purposes of this filter, a RMCP secondary TCP packet is defined to be any packet that meets all of the following requirements: <ul style="list-style-type: none"> The Ethernet <i>Destination Address</i> field contains the MAC address assigned to the MC. The Ethernet <i>Type</i> field contains 0x86DD (IPv6) Or 0x0800 (IPv4). The IP <i>Destination Address</i> field contains the one of the IP address assigned to the MC. The IP <i>Protocol</i> field contains 6 (TCP). The TCP <i>Destination Port</i> field contains 664 [secure aux bus (secondary RMCP port)].
31:11	Reserved	None.

9.6.4.14.5.1 Enable Unicast Infrastructure Filter Response

The NC, in the absence of a checksum error or identifier mismatch, always accept the Enable Unicast Infrastructure Filter command and send a response using the format listed in [Table 9-52](#). Currently no command-specific reason codes are identified for this response.

Table 9-52. Enable Unicast Infrastructure Filter Response Packet Format

Bits			
00..15	NC-SI Header		
16..19	Response Code	Reason Code	
20..23	Manufacturer ID (Intel 0x157)		
24..27	0x25	0x4	Reserved
28..29	Checksum		

9.6.4.14.6 Get Shared IP Capabilities Command (Intel Command 0x25, index = 0x5)

An MC uses the Get Shared IP Capabilities command to determine the level of support of shared IP of the device. The format of a Get Shared IP Capabilities command packet is listed in [Table 9-53](#).



Table 9-53. Get Shared IP Capabilities Command Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0x5	Reserved	
24..27	Checksum			

9.6.4.14.6.1 Get Shared IP Capabilities Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Get Shared IP Capabilities command and send a response, using the format listed in Table 9-54. Currently no command-specific reason codes are identified for this response.

Table 9-54. Get Shared IP Capabilities Response Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x5	Number of Mixed IP address	Number of IPv4 only addresses
28..31	Number of IPv6 only addresses	Number of Ports	Number of bindings	Filtering capabilities
32..35	Unicast Infrastructure Filter capabilities			
36..30	Checksum			

- Number of mixed IP addresses — The number of supported IP filters that can be used for IPv4 or IPv6. The X710-TM4 does not support mixed IP address filters.
- Number of IPv4 only addresses — The number of supported IP filters that can be used for IPv4 only. The X710-TM4 supports three IPv4 address filters.
- Number of IPv6 only addresses — The number of supported IP filters that can be used for IPv6 only. The X710-TM4 supports four IPv6 address filters.
- Number of ports — The number of supported port filters.
- Number of bindings — Defines the number of IP addresses that can be bound with different ports.
- Unicast infrastructure filter capabilities — Defines the optional unicast infrastructure filter capabilities that the channel supports. The bit definitions for this field correspond directly with the bit definitions for the *Unicast Infrastructure Filter Settings* field defined for the Unicast Infrastructure Filter command listed in Table 9-51. A bit set to 1b indicates that the channel supports the filter associated with that bit position; otherwise, the channel does not support that filter. The X710-TM4 supports all filters but ICMPv6 filtering, so the returned value is 0x7FB.
- Table 9-55 lists the bits fields in the *Filtering Capabilities* field.

**Table 9-55. Filtering Capabilities Field**

Bit Position	Field Description	Value Description
0	IPv4 support	0b = IPv4 filtering is not supported. 1b = IPv4 filtering is supported.
1	IPv6 support	0b = IPv6 filtering is not supported. 1b = IPv6 filtering is supported.
2	Protocol filtering support	0b = Filtering by protocol is not supported. 1b = Filtering by protocol is supported.
3	Source port filtering support	0b = Port filtering is supported only for destination port. 1b = Port filtering is supported for destination port or source port.
7:4	Reserved	Reserved.

9.6.4.14.7 Shared IP Enable Broadcast Filtering Command (Intel Command 0x25, Index = 0x6)

A new shared IP enable broadcast filtering is defined to enable the MC to limit the flow of ARP requests to those that contain a target IP address value that matches the MC IP address.

This command should be used instead of the regular NC-SI Enable Broadcast Filtering command.

Note: Receiving a standard NC-SI Enable Broadcast Filtering command enables the matching bits in this command. Receiving a standard NC-SI Disable Broadcast Filter Command clears the settings in this command.

The format of an Shared IP Enable Broadcast Filtering command packet is listed in [Table 9-50](#).

Table 9-56. Shared IP Enable Broadcast Filtering Command

Bytes	Bits			
	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x6	Reserved	
28..31	Shared IP Broadcast Packet Filter Settings			
32..35	Checksum			
36..29	Padding			

The content of the *Shared IP Broadcast Packet Filter Settings* field is listed in [Table 9-57](#). Bit 4 has been added to the standard enable broadcast filtering command limit ARP broadcast packets to the MC IP address.



Table 9-57. Shared IP Broadcast Packet Filter Settings Field

Bit Position	Field Description	Value Description
3:0	As defined in DSP0222	As defined in DSP0222 in 8.4.33 Enable Broadcast Filter Command (0x10) - table 68
4	Limit ARP Broadcast Packets to Management Controller IP Address.	When bit 0 is set, it limits the flow of ARP packets to the MC as follows: 0x1 = Forward only ARP broadcast packets that are targeted at IP addresses bound to the MC. 0x0 = Forward all ARP broadcast packets to the MC. All the IPs set by the Set IP command before this command is given will be included in forwarding. This field is optional. If unsupported, the behavior for ARP packets is set according to bit 1 in this structure. The value must be set to 0b if unsupported.
31:5	Reserved	None.

9.6.4.14.7.1 Shared IP Enable Broadcast Filtering Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Shared IP Enable Broadcast Filtering command and send a response using the format listed in [Table 9-52](#). Currently no command-specific reason codes are identified for this response.

Table 9-58. Shared IP Enable Broadcast Filtering Packet Format

	Bits		
00..15	NC-SI Header		
16..19	Response Code	Reason Code	
20..23	Manufacturer ID (Intel 0x157)		
24..27	0x25	0x6	Reserved
28..29	Checksum		

9.6.4.14.8 Shared IP Enable Global Multicast Filtering Command (Intel Command 0x25, Index = 0x7)

A new shared IP enable global multicast filtering is defined to enable the MC to enable the forwarding of IEEE 802.1X Extensible Authentication Protocol over LAN (EAPOL) frames to the MC IP address. IEEE 802.1X defines methods for port-based network access control.

This command should be used instead of the regular NC-SI Enable Global Multicast Filtering command.

Note: Receiving a standard NC-SI Enable Global Multicast Filtering command enables the matching bits in this command. Receiving a standard NC-SI Disable Global Multicast Filter command clears the settings in this command.

The format of an Shared IP Enable Global Multicast Filtering command packet is listed in [Table 9-59](#).

**Table 9-59. Shared IP Enable Global Multicast Filtering Command**

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x7	Reserved	
28..31	Shared IP Multicast Packet Filter Settings			
32..35	Checksum			
36..29	Padding			

The content of the *Shared IP Multicast Packet Filter Settings* field is listed in [Table 9-60](#). Bit 4 has been added to the standard enable broadcast filtering command limit ARP broadcast packets to MC IP address.

Table 9-60. Shared IP Multicast Packet Filter Settings Field

Bit Position	Field Description	Value Description
0:2	As defined in DSP0222	As defined in DSP0222 in 8.4.37 Enable Global Multicast Filter Command (0x12) - table 74.
3	IEEE 802.1X EAPOL	0x1 = Forward this packet type to the MC. 0x0 = Filter out this packet type. For the purposes of this filter, a IEEE 802.1X multicast packet is defined to be any packet that meets all of the following requirements: <ul style="list-style-type: none"> The destination MAC address field is set to the layer 2 multicast address 01:80:c2:00:00:03. The EtherType field is set to 0x888E (802.1X PAE). This field is optional. If unsupported, multicast 802.1X packets are blocked when multicast filtering is enabled, unless they are matched by an address filter configured using the Set MAC Address command. The value must be set to 0b if unsupported.
4:31	Reserved	None.

9.6.4.14.8.1 Shared IP Enable Global Multicast Filtering Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Shared IP Enable Global Multicast Filtering command and send a response using the format listed in [Table 9-61](#). Currently no command-specific reason codes are identified for this response.

Table 9-61. Shared IP Enable Global Multicast Filtering Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x7	Reserved	
28..29	Checksum			



9.6.4.14.9 Get Shared IP Parameters Command (Intel Command 0x25, Index = 0x8)

The Get Shared IP parameters command can be used by the MC to request that the channel send the MC a copy of part of the currently stored parameter settings that have been put into effect by the MC related to shared IP filtering. The format of a Get Shared IP Capabilities command packet is listed in Table 9-62.

Table 9-62. Get Shared IP Parameters Command Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0x8	Reserved	
24..27	Checksum			

9.6.4.14.9.1 Get Shared IP Parameters Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Get Shared IP parameters command and send a response using the format listed in Table 9-63. Currently no command-specific reason codes are identified for this response.

Table 9-63. Get Shared IP Parameters Response Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x5	Reserved	
28..31	Number of IP Addresses	IP addresses Flags		
32..35	Number of ports	Ports Flags		
36..39	Unicast Infrastructure Filter Settings			
40..43	Broadcast Filtering Settings			
44..47	Multicast Filtering Settings			
48..51	Checksum			

- Number of IP addresses — The number of supported IP filters including all the types of IP addresses (IPv4 only, IPv6 only and mixed).
- IP address flags — The enable/disable state for each supported IP address. See Table 9-64.

**Table 9-64. IP Address Flags Field**

Bit Position	Field Description	Value Description
0	IP Address 1 Status	0b = Default or unsupported or disabled. 1b = Enabled.
1	IP Address 2 Status Or Reserved	0b = Default or unsupported or disabled. 1b = Enabled.
2	IP Address 3 Status Or Reserved	0b = Default or unsupported or disabled. 1b = Enabled.
...		
23	IP Address 24 Status Or Reserved	0b = Default or unsupported or disabled. 1b = Enabled.

Note: IP address flags are organized in the following order: IPv4 addresses first, followed by IPv6 addresses, followed by mixed addresses, with the number of each corresponding to those reported through the Get Shared IP Capabilities command.

For example, if the interface reports four IPv4 filters, two IPv6 filters, and two mixed filters, then IP addresses 1 through 4 are those currently configured through the interface's IPv4 filters, IP addresses 5 and 6 are those configured through the IPv6 filters, and 7 and 8 are those configured through the mixed filters.

The actual settings of each enabled IP address can be found using the Get IP address command

- Number of ports — The number of supported port filters.
- Port flags — The enable/disable state for each supported ports. See [Table 9-64](#).

Table 9-65. Port Flags Field

Bit Position	Field Description	Value Description
0	Port 1 status	0b = Default or unsupported or disabled. 1b = Enabled.
1	Port 2 status or Reserved	0b = Default or unsupported or disabled. 1b = Enabled.
2	Port 3 status or Reserved	0b = Default or unsupported or disabled. 1b = Enabled.
...		
23	Port 24 status or Reserved	0b = Default or unsupported or disabled. 1b = Enabled.

Note: The actual settings of each enabled port can be found using the Get Port command

- Unicast Infrastructure Filter Settings — Defines the optional unicast infrastructure filter capabilities settings. The bit definitions for this field correspond directly with the bit definitions for the *Unicast Infrastructure Filter Settings* field defined for the Unicast Infrastructure Filter command in [Table 9-51](#). A bit set to 1b indicates that the filter associated with that bit position is enabled; otherwise, the filter is not enabled.
- Broadcast Filter Settings — Defines the optional broadcast filter settings. The bit definitions for this field correspond directly with the bit definitions for the *Broadcast Filter Settings* field defined for the Shared IP Broadcast Filtering command in [Table 9-57](#). A bit set to 1b indicates that the filter associated with that bit position is enabled; otherwise, the filter is not enabled.



- Global Multicast Filter Settings — Defines the optional multicast filter capabilities settings. The bit definitions for this field correspond directly with the bit definitions for the *Multicast Filter Settings* field defined for the Shared IP Global Multicast Filtering command in [Table 9-57](#). A bit set to 1b indicates that the filter associated with that bit position is enabled; otherwise, the filter is not enabled.

9.6.4.14.10 Set Binding Command (Intel Command 0x25, Index = 0x9)

The Set Binding command is used by the MC to define which combination of MAC addresses, VLAN tags, IP addresses and TCP/UDP ports should be forwarded to the MC. The format of a Set Binding command packet is listed in [Table 9-66](#).

Once a Set Binding command is activated, all the previous forwarding rules based on the Set MAC Address or Set VLAN filter commands are disabled and should be re-enabled using the Set Binding command. Subsequent Set MAC Address or Set VLAN filter commands are used to enable MAC or VLAN addresses for the Set Binding command but does not impact the forwarding rules.

Table 9-66. Set Binding Command Packet Format

Bytes	Bits			
	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0x9	Binding Index	Set Binding Flags
24..27	Enabled MAC addresses			
28..31	Enabled VLAN			
32..35	Enabled IP addresses			
36..39	Enabled Ports (MSB)			
40..43	Enabled Ports (LSB)			
44..47	Checksum			

[Table 9-67](#) lists the fields in the *Set Binding Flags* field.

Table 9-67. Set Binding Flags Field Descriptions

Bit Position	Field Description	Value Description
0	Enable	0b = Disable the binding 1b = Enable the binding
1	Exclusive to MC	0b = Traffic matching this filter is sent to the MC and to the host 1b = Traffic matching this filter is sent to the MC only.
2	Apply to network ¹	0b = Do not compare traffic received from the network when checking this binding. 1b = Compare traffic received from the network when checking this binding.
3	Apply to host	0b = Do not compare traffic received from the host when checking this binding. 1b = Compare traffic received from the host when checking this binding.
7:4	Reserved	Reserved.

1. At least one of the apply to network/host flags should be set for enabled bindings. Clearing both of them is equivalent to disabling the filter.



- Binding Index — Indicates which binding is configured by the command. The value should be smaller than the number of supported bindings as reported in the get shared IP capabilities response in the *Number of Bindings* field.
- Enabled MAC Addresses — The MAC addresses participating in this binding. The numbering of the MAC addresses is similar to the one used in the MAC address flags in the get parameters response. Namely, MAC addresses are returned in the following order: unicast filtered addresses first, followed by multicast filtered addresses, followed by mixed filtered addresses, with the number of each corresponding to those reported through the Get Capabilities command. A MAC address can be added to a binding only if previously enabled through a Set MAC Address NC-SI command
- Enabled VLAN — The VLAN IDs participating in this binding. The numbering of the VLAN IDs. A VLAN tag can be added to a binding only if previously enabled through a Set VLAN Filter NC-SI command.
- Enabled IP Addresses — The IP addresses participating in this binding. The numbering of the IP addresses is similar to the one used in [Section 9.6.4.14.9](#). An IP address can be added to a binding only if previously enabled through a Set IP Address Intel OEM command.
- Enabled Ports — The ports participating in this binding. A port can be added to a binding only if previously enabled through a Set Port Intel OEM command.

9.6.4.14.10.1 Set Binding Address Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Set Binding command and send a response using the format listed in [Table 9-68](#).

Table 9-68. Set Binding Response Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0x9	Reserved	
28..31	Checksum			

9.6.4.14.11 Get Binding Command (Intel Command 0x25, Index = 0xA)

A MCr uses the Get Binding command to determine the current programming of one of the bindings in a NC. The format of a *Get Binding* command packet is listed in [Table 9-69](#).

Table 9-69. Get Binding Command Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0xA	Binding Number	Reserved
24..27	Checksum			



- Binding Index: Indicates which binding is requested by the command. The value should be smaller than the number of supported bindings as reported in the Get Shared IP Capabilities Response in the *Number of Bindings* field.

9.6.4.14.11.1 Get Binding Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Get Binding command and send a response using the format listed in [Table 9-70](#).

Table 9-70. Get Binding Response Packet Format

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0xA	Binding Number	Get Binding flags
28..31	Enabled MAC addresses			
32..35	Enabled VLAN			
36..39	Enabled IP addresses			
40..43	Enabled Ports (MSB)			
44..47	Enabled Ports (LSB)			
48..51	Checksum			

The fields in the Get Binding response are equivalent to their counterparts in the Set Binding command.

9.6.4.14.12 Set Shared Mode Command (Intel Command 0x25, Index = 0xB)

An MC uses the Set Shared Mode command to indicate to the NIC it intends to operate in shared MAC/IP mode or in dedicated MAC mode.

If used, this command should be sent before any of the regular or OEM NC-SI commands used to set forwarding filters. When this command is received, all the filters are cleared.

This command is only needed when the Intel OEM commands with command ID 0x25 are used to configure the shared behavior. If other commands are used, users should take care of the right configuration of the filters.

When shared mode is activated, the Set MAC and Set VLAN NC-SI commands do not impact the receive filtering until a Set Binding or Enable Unicast Infrastructure Filter command is received.

Any other command from this section (9.6.4.14) received before shared mode is set fails with a Not is Shared Mode (0x5092) reason.

The format of a Set Shared Mode command packet is listed in [Table 9-71](#).

**Table 9-71. Set Shared Mode Command Packet Format**

Bytes	Bits			
	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x25	0xB	Shared Mode	Reserved
24..27	Checksum			

- Shared mode:
 - 0x0: Dedicated MAC mode.
 - 0x1: Shared MAC/IP mode.

9.6.4.14.12.1 Set Shared Mode Response

The NC must, in the absence of a checksum error or identifier mismatch, always accept the Set Shared Mode command and send a response using the format listed in [Table 9-72](#).

Table 9-72. Set Shared Mode Response Packet Format

Bytes	Bits			
	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x25	0xB	Shared Mode	Reserved
48..51	Checksum			

9.6.4.15 OS2BMC Configuration

These commands control enabling of the OS2BMC flow.

9.6.4.15.1 Enable OS2BMC Flow Command (Intel Command 0x40, Index 0x1)

Bytes	Bits			
	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x40	0x01		



9.6.4.15.2 Enable OS2BMC Flow Response (Intel Command 0x40, Index 0x1)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x40	0x01		

9.6.4.15.3 Enable OS-to-BMC Flow command (Intel Command 0x40, Index 0x2)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x40	0x02		

9.6.4.15.4 Enable Network-to-BMC Flow Response (Intel Command 0x40, Index 0x2)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x40	0x02		



9.6.4.15.5 Enable Both Host and Network-to-BMC Flows Command (Intel Command 0x40, Index 0x3)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...21	0x40	0x03		

9.6.4.15.6 Enable Both Host and Network-to-BMC Flows Response (Intel Command 0x40, Index 0x3)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x40	0x03		



9.6.4.15.7 Set MC IP Address Command (Intel Command 0x40, Index 0x4)

This command is used to expose the MC IP address to the host. This command is supported by the X710-TM4, but no action is taken upon reception. The IP address is not stored and when a Get OS2BMC Parameters command is received, the IP valid flag is cleared.

The IP type entry indicate whether the IP address is an IPv4 or an IPv6 address:

- 0 = IPv4.
- 1 = IPv6.
- 2 = No IP address, then the command should not include an IP address.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x40	0x04	IP type	IPv6 Address (MSB, byte 15)/IPv4 Address (MSB, byte 3)
24..27	IPv6 Address (byte 14)/IPv4 Address (byte 2)	IPv6 Address (byte 13)/IPv4 Address (byte 1)	IPv6 Address (byte 12)/IPv4 Address (LSB, byte 0)	IPv6 Address (byte 11)/Reserved
28..31
32..35
36..38	IPv6 Address (LSB, byte 0)/Reserved	

9.6.4.15.8 Set BMC IP Address Response (Intel Command 0x40, Index 0x4)

	Bits			
Bytes	31...24	23...16	15...08	07...00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...25	0x40	0x04		



9.6.4.15.9 Get OS2BMC Parameters Command (Intel Command 0x41)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20	0x41			

9.6.4.15.10 Get OS2BMC Parameters Response (Intel Command 0x41)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x41	Status	IPv6 Address (MSB, byte 15)/IPv4 Address (MSB, byte 3)	IPv6 Address (byte 14)/IPv4 Address (byte 2)
28..31	IPv6 Address (byte 13)/IPv4 Address (byte 1)	IPv6 Address (byte 12)/IPv4 Address (LSB, byte 0)	IPv6 Address (byte 11)/Reserved	..
32..35
36..39
39..40	..	IPv6 Address (LSB, byte 0)/Reserved		

Where the status byte partition is as follows:



Table 9-73. Status Byte Description

Bits	Content
0	0b = IPv4. 1b = IPv6. Relevant only if the IP address valid bit is set.
1	IP address valid. Never valid for the X710-TM4.
1:0	Reserved.
2	Network to MC Status. 0b = Network-to-BMC flow is disabled. 1b = Network-to-BMC flow is enabled.
3	OS2BMC Status. 0b = OS2BMC flow is disabled. 1b = OS2BMC flow is enabled.
7:4	Reserved.

9.6.4.16 Diagnostic Command (Intel Command 0x48)

9.6.4.16.1 Get Controller Information Command (Intel Command 0x48, Index 0x1)

This command gathers the controller identification information and return it back to the MC.

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20...23	0x48	0x1		



9.6.4.16.2 Get Controller Information Response (Intel Command 0x48, Index 0x1)

	Bits			
Bytes	31:24	23:16	15:08	07:00
00...15	NC-SI Header			
16...19	Response Code		Reason Code	
20...23	Manufacturer ID (Intel 0x157)			
24...27	0x48	0x01	Reserved	Number of Inventory entries
28...31	Controller Info Item 1 ID	Controller Info Item 1 length	Controller Info Item 1 Data	
...			
...	Controller Info Item 2 ID	Controller Info Item 2 length	Controller Info Item 2 Data	
...			
...	Controller Info Item n ID	Controller Info Item n length	Controller Info Item n Data	
...			

The possible inventory items are described as follows. Note that not all the inventory items would be present in all the implementations of this command.

Table 9-74. Controller Information Items

ID	Length (in bytes)	Data	Notes
0x00	3	Device ID (2 bytes) + RevID	This is the hardware default value (no value programmed via the NVM).
0x0B	2	NVM Image Version	
0x0C	4	EMP ROM Internal Version	
0x0D	4	EMP Flash Internal version	Same version as in Get Version Admin command.
0x0E	2	PXE firmware version	MajorVersion.MinorVersion.Build.
0x0F	2	iSCSI firmware version	
0x10	2	uEFI firmware version	
0x16	2	Reserved	
0x20	<=32	Serial Number	VPD-RO keyword 'SN' or, if not available, the first entry of "MAC addresses" field under "SW-Data-Recovery" NVM section encoded to hexadecimal upper-case ASCII string.
0x21	<=32	Serial Number	VPD-RO keyword 'PN' or, if not available, the "PBA" field under "SW-Data-Recovery" NVM section.

9.6.4.17 Initialization Error AEN (Intel AEN 0x82)

Note: This AEN was previously called NVM error AEN.



The following is the AEN that might be sent by the NC following a detection of a wrong CRC/checksum on a firmware related section in the NVM. NC is required to store this AEN internally until a connection to the MC is established so the report could be issued in all cases.

This AEN must be enabled using the NC-SI AEN Enable command, using bit 18 (0x40000) of the AEN enable mask.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00...15	NC-SI AEN Header			
20...23	Reserved			0x82
24	Index of NVM module on which the error was found. The encoding is as defined in <i>GL_MNG_FWSM.EXT_ERR_IND</i> field.			

9.7 Management Component Transport Protocol (MCTP)

9.7.1 Overview

MCTP defines a communication model intended to facilitate communication between:

- MCs and other MCs
- MCs and management devices

The communication model includes a message format, transport description, message exchange patterns, and configuration and initialization messages.

The basic MCTP specification is described in DMTF's DSP0236 document.

MCTP is designed so that it can potentially be used on many bus types. The protocol is intended to be used for intercommunication between elements of platform management subsystems used in computer systems, and is suitable for use in mobile, desktop, workstation, and server platforms.

Currently, specifications exist for MCTP over PCIe (DMTF's DSP0238) and over SMBus (DMTF's DSP0237). A specification for MCTP over USB is also planned.

MCs such as a Baseboard Management Controller (BMC) can use this protocol for communication between one another, as well as for accessing management devices within the platform.



9.7.1.1 NC-SI Over MCTP

MCTP is a transport layer protocol that does not include the functionality required to control the PT traffic required for an MC connection to the network. This functionality is provided by encapsulating NC-SI traffic as defined in DMTF's DSP0222 document.

The details of NC-SI over MCTP protocol are defined in the DMTF's DSP0261 - NC-SI Over MCTP Specification.

An NC-SI over MCTP implementation guide can be found in the DMTF's DSP0219 white paper.

The NC-SI over MCTP specification defines two types of MCTP message types: NC-SI (0x2) and Ethernet (0x3). The X710-TM4 supports both messages. When used only for control, then only the NC-SI (0x2) message type is supported.

In addition to the previous message types supported by the X710-TM4, the PCIe based VDM message type is also supported over PCIe to support ACL commands.

Details of the NC-SI over MCTP can be found in [Section 9.7.5](#).

Enabling NC-SI over MCTP for pass through traffic is done by setting the *Redirection Sideband Interface* field in the *Common Manageability Parameters* NVM word to "MCTP over PCI and SMBus".

Enabling NC-SI over MCTP for control traffic is done by setting the *Control Interface* field in the *Common Manageability Parameters* NVM word to "MCTP over SMBus/PCI" and by setting the *NC-SI* bit in *Common Manageability Parameters 2* NVM word.

9.7.1.2 MCTP Usage Model

The X710-TM4 supports NC-SI over MCTP protocol over the PCIe and SMBus buses. The X710-TM4 can connect through MCTP to a MC or the ME engine in the chipset as described in [Figure 9-9](#).

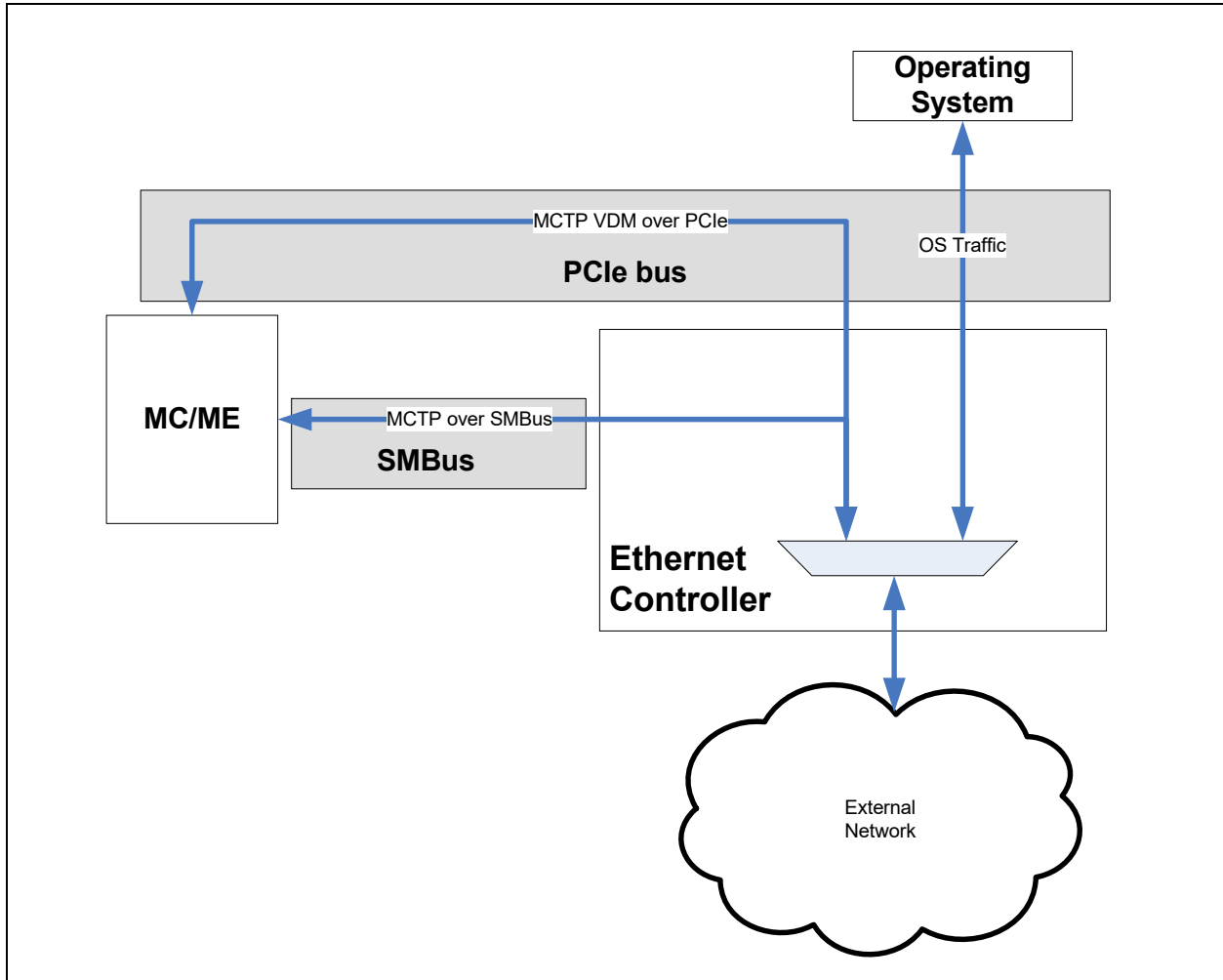


Figure 9-9. the X710-TM4 MCTP Connections

9.7.2 NC-SI to MCTP Mapping

The four network ports of the X710-TM4 (mapped to two NC-SI channels) are mapped to a single MCTP endpoint on SMBus and to another endpoint over PCIe.

The PCIe endpoint is mapped to a PCIe requester ID according to the following flow:

1. If the *Bus Master Enable* bit of at least one of the functions is set, the endpoint is mapped to function the first available function.
2. If the *Bus Master Enable* bits of all functions are cleared, the MCTP endpoint on PCIe is not exposed and the MCTP traffic is routed through the SMBus endpoint.

The slave address used for the SMBus endpoint is the slave address of the first port.

Section 9.7.2.1 describes the transition between the two buses.



Both endpoints (SMBus and PCIe) might be active concurrently. However, PT traffic might be transferred only through one of them. If the PCIe endpoint is active, it is used for PT traffic; otherwise, the SMBus endpoint is used. The Set EID command can be used to force the transition for the PCIe endpoint to the SMBus endpoint if the bus owner determines the PCIe channel is not functional.

For each channel (SMBus or PCIe), the X710-TM4 should expect MCTP commands from two sources: the bus owner and the MC. In addition, it should expect PT traffic through one interface only. Thus, it should be able to process up to five interleaved commands/data:

- An MCTP control/OEM command from the PCIe bus owner (single packet message).
- An MCTP control/OEM command from the SMBus bus owner (single packet message).
- An MCTP control/OEM command from the MC over SMBus (single packet message).
- An MCTP control/OEM command from the MC over PCIe (single packet message).
- An NC-SI command or Ethernet packet from the MC over the active channel.

A single source should not interleave packets it sends.

The topology used for MCTP connection is shown in [Figure 9-10](#).

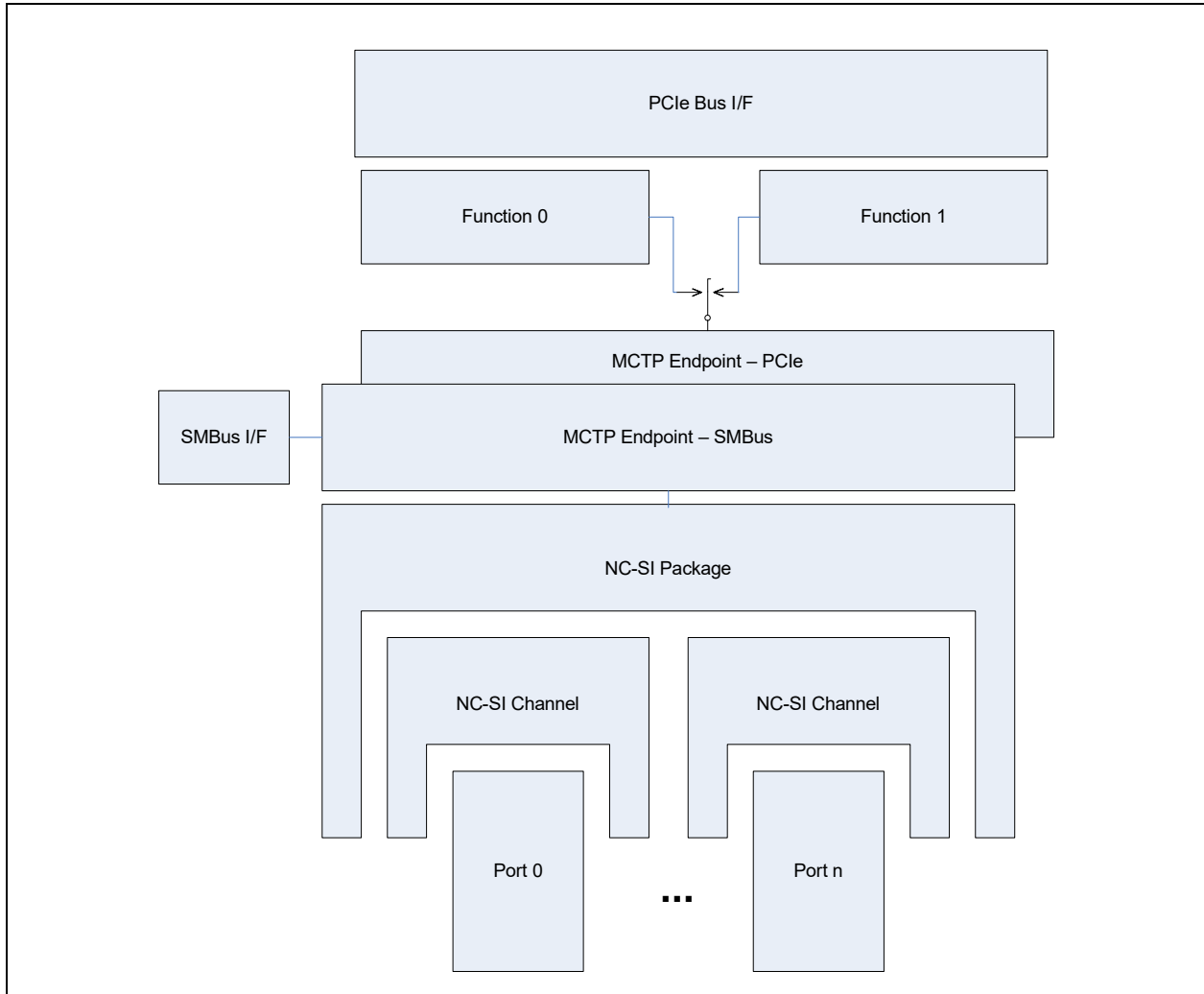


Figure 9-10. MCTP Endpoints Topology

9.7.2.1 Detecting an MC EID and Physical Address

In order to enable transactions between the MC and the NIC, the bus physical address (SMBus or PCIe) and the EID of the partner needs to be discovered. NICs don't try to discover the MC and assume the MC initiates the connection. If the NIC is in an NC-SI initial state, then the EID and the physical address of the MC are extracted from the Clear Initial State command parameters or any other NC-SI command received later with a channel ID of the X710-TM4. Subsequent PT traffic is received from or sent to this address only.

If the EID or the physical address of the NIC changes, it indicates the changes to bus owner so that the routing tables can be updated. There is no attempt to directly send an indication to the MC about the change.

See more details in next section.



9.7.2.2 Bus Transition

The following section defines the transition flow between PCIe and SMBus as the bus on which MCTP flows. Figure 9-11 shows the flow to transition between PCIe and SMBus. The following parameters are used to define the flow:

- NIC EID on PCIe
- NIC EID on SMBus
- NIC PCIe Target ID
- Bus Owner EID on PCIe
- Bus Owner EID on SMBus
- Bus Owner PCIe Target ID
- Bus Owner SMBus Address
- MC EID on PCIe
- MC EID on SMBus
- MC PCIe Target ID
- MC SMBus Address
- NIC SMBus Address

All these variables are initialized to zero at power on apart from the SMBus address of the endpoint (NIC), which might be initialized from a NVM value.

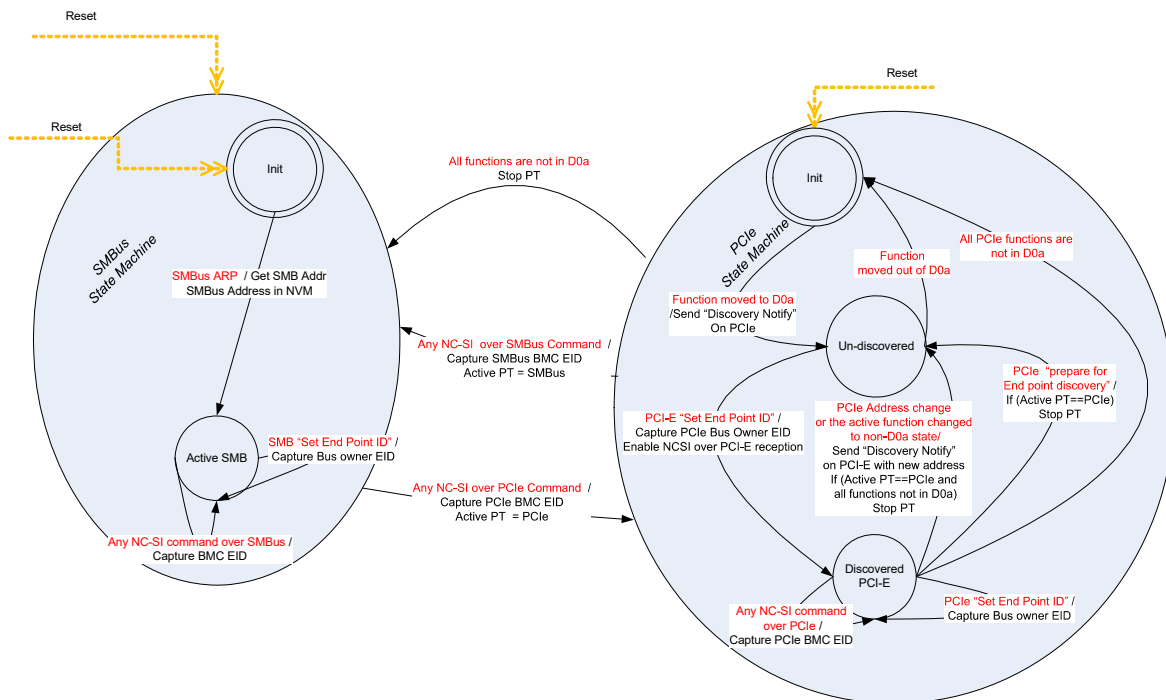


Figure 9-11. MCTP Bus Transition State Machine



9.7.2.2.1 Initial Assignment Flow

- At power on, the NIC or MC MCTP channel is connected to the SMBus, is not assigned an EID and is in an undiscovered state.
- The bus owner might preform an SMBus ARP cycle to assign an SMBus address to the NIC or to the MC. Otherwise, a fixed address might be used. It is assumed that the SMBus address does not change after initialization time.
- The bus owner performs an EID assignment using a Set Endpoint ID MCTP command. The NIC or the MC captures the SMBus address of the bus owner from the *SMBus Source Slave address* field, the bus owner EID from the *Source Endpoint ID* field and the NIC/MC EID from the *Destination Endpoint ID* field in the MCTP header as described in section 10.3 of DSP0236. The NIC/MC is now in a discovered state.
- The MC might detect the NIC EID using one of the two following modes:
 - Static configuration of the NIC SMBus address in the MC database and Get Routing Table Entries command to find the EID matching the SMBus address.
 - Get all endpoints through a Get Routing Table Entries command and find endpoints supporting NC-SI using the Get Message Type Support command for each endpoint.
- Once the NIC is found, the MC might send a Clear Initial State command to the NIC to start the NC-SI configuration. The NIC captures the MC SMBus address and MC EID from any NC-SI command received.
- After the NC-SI channels are enabled, traffic might be sent using the MC and NIC addresses previously discovered.
- The MC might also send a Get UUID command to get a unique identifier of the NIC that might be used later for re-connection upon topology changes.

9.7.2.2.2 SMBus to PCIe Transition

- If the NIC or the MC detects that the PCIe bus is available by detecting a function that moved to D0a state, it might request a transition using a Discovery Notify MCTP command on the PCIe bus. This command should be sent with a route to root-complex addressing as described in DSP0238 section 6.8. The source EID should be the EID previously assigned on the SMBus.
- After receiving the Discovery Notify MCTP command on the PCIe bus, the bus owner sends a Set Endpoint ID MCTP command on the PCIe bus and updates the routing table. The bus owner might choose to wait for the Discovery Notify MCTP command of both the MC and the NIC to do the transition. The bus owner should try to keep the EID previously assigned on the SMBus as the EID on PCIe bus.
- After receiving the Set Endpoint ID MCTP command, the NIC waits for an NC-SI command from the MC indicating it is ready to transition the connection to PCIe. After receiving such a command, the NIC transitions its PT traffic to the PCIe bus using the newly received addresses.
- The MC on its side, needs to discover the PCIe address of the NIC. This can be done using the Resolve Endpoint ID command if only the physical address changed or using the Resolve Endpoint UUID command also if both EID and physical address changed. It can then send an NC-SI command to the NIC to initiate the transition. The MC should not send any PT packets from the moment it sent the first NC-SI command on the PCIe and the moment a response is received for this command.
- The transition of NC-SI traffic (PT or commands/responses) from SMBus to PCIe should be done on a packet boundary and should not interrupt a packet fragmentation or reassembly.



9.7.2.2.3 PCIe Target ID Change

The target ID of one of the endpoints might change, either due to a new enumeration of the PCIe bus or due to the disabling of one of the functions in the device (move to a non D0a state). In this case, the following flow should be used:

- The endpoint should send a Discovery Notify MCTP command on the PCIe bus using the new Requester ID.
- After receiving the Discovery Notify MCTP command with the new requester ID, the bus owner sends a Set Endpoint ID MCTP command on the PCIe bus and updates the routing table. The bus owner should try to keep the EID previously assigned on the SMBus as the EID on the previous requester ID.
- The bus owner sends an Routing Information Update command to all supporting endpoints that might then update the parameters of their counterpart they use.

9.7.2.2.4 PCIe to SMBus Transition

- If the NIC or the MC detects that the PCIe bus is not available by detecting a transition of all functions to a non D0a state, it stops using the PCIe for PT traffic or NC-SI traffic.
- Upon detection of the unavailability of the PCIe bus, the MC transitions the NC-SI channel to the MCTP over SMBus as previously described.
- The transition of NC-SI traffic (PT or commands/responses) from PCIe to SMBus might done at any stage and might interrupt a packet fragmentation or reassembly, as it is assumed that such a transition occurs only when the PCIe bus is not available anymore.

9.7.3 MCTP over PCIe

9.7.3.1 Message Format

The message format used for NC-SI over MCTP over PCIe is as follows:

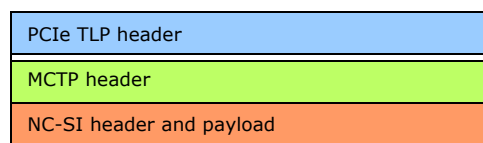


Table 9-75. NC-SI/Ethernet Over MCTP Over PCIe Message Format

+0								+1								+2								+3								
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
FMT 011				Type 10r2r1r0 ¹				R	TC 000				R	A	r ²	R	T	H ²	T	D ²	E	P ²	Attr [1:0] 2	AT 00	Length 00_000x_xxxx							
PCI Requester ID																PCI Tag Field						Message Code Vendor Defined = 0111_1111b										
																R	Pad Len	MCTP VDM code - 0000b														



Table 9-75. NC-SI/Ethernet Over MCTP Over PCIe Message Format

+0								+1								+2								+3							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
PCI Target ID (For Route by ID messages, otherwise = Reserved)																Vendor ID = 0x1AB4 (DMTF)															
MCTP Reserved				Header version = 1				Destination endpoint ID								Source endpoint ID								S	E	SEQ#	T	Tag			
I	Message Type = 0x02/0x03							NC-SI Command/Pass Through data																							
C																														
NC-SI Command/Pass Through data																															

1. r2r1r0 = .
000b: Route to Root Complex.
010b: Route by ID.
011b: Broadcast from Root Complex.
2. TD = 0, EP = 0, TH = 0, Attr[2:0] = 0 for sent packets and is ignored for received packets.

9.7.3.2 PCIe Discovery Process

The X710-TM4 follows the discovery process described in section 5.9 of the MCTP PCIe VDM Transport Binding Specification (DSP0238).

After receiving an endpoint discovery message (while in undiscovered stage), the X710-TM4 exposes the endpoint on the selected function as previously described.

If the selected function moves to D0u after the endpoint was discovered, or if the bus number of the X710-TM4 changes due to a re-enumeration of the bus, the X710-TM4 sends a discovery notify message to indicate to the MC that it should do a re-enumeration of the device to discover the new endpoint.

9.7.3.3 MCTP Over PCIe Special Features

The X710-TM4 supports the following optional features of MCTP when running over PCIe:

1. Rate limiting
2. ACLs



9.7.3.3.1 MCTP Uplink Rate Limiting

As the PCIe link can carry a traffic bandwidth much higher than what the MC can sustain, in order to avoid drop of packets, the X710-TM4 allows rate limiting of the MCTP PT traffic. The X710-TM4 supports rate limiting between 1 Mb/s and 1 Gb/s. The following parameters define the behavior of the rate limiter:

- Max rate limit (fixed from NVM via the MCTP rate in the MCTP rate limiter config 1 word).
- The max burst size (fixed from NVM via the *MCTP max credits* field in the in the MCTP rate limiter config 2 word). To limit the max burst to one VDM, set this parameter to 5.
- Decision point (fixed from NVM via the *decision point* field in the in the MCTP rate limiter config 2 word).

9.7.3.3.2 ACLs

The X710-TM4 supports a set of ACLs that allows reception of sensitive commands only from a specific bus number (in the requester ID). The device and function part of the requester ID are ignored for this purpose.

If ACLs are enabled (by clearing the Disable ACLs NVM bit) the following flow is used decide which packets are accepted.

Commands can be divided to three types:

1. ACL programming commands: Such commands can be received only from the address that sent the [Prepare For Endpoint Discovery command via broadcast routing](#).
2. Sensitive commands including all the NC-SI commands and PT traffic. These commands can be received only from requesters whose bus number is set in the ACL list. If an MCTP packet is dropped, then the SPMEACLD counter is increased. This counter can be read by the MCTP bus owner using the Get ACL Violation Counters command.
3. Regular MCTP commands are received from any requester; however, the Set EID command is processed only if received from the address that sent the Prepare For Endpoint Discovery command via broadcast routing.

The X710-TM4 supports 4 ACL entries.

Note: This feature is disabled by default in NVM as it requires a system infrastructure supporting the feature.



9.7.4 MCTP Over SMBus

The message format used for NC-SI over MCTP over SMBus is as follows:

SMBus header/PEC
MCTP header
NC-SI header and payload

Table 9-76. NC-SI/Ethernet Over MCTP Over SMBus Message Format

+0								+1								+2								+3								
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
Destination Slave Address								0	Command Code = MCTP = 0Fh								Byte count								Source Slave Address							1
MCTP Reserved				Header version = 1				Destination endpoint ID								Source endpoint ID								S	E	SEQ#	T	Tag				
I	C	Message Type = 0x02/0x03						NC-SI Command/Pass Through data																								
1																															
NC-SI Command/Pass Through data																																
PEC																																

- IC = 0 for pass through sent packets, or as defined by Firmware for other packets. Packets received with IC = 1 are dropped..

9.7.4.1 SMBus Discovery Process

The X710-TM4 follows the discovery process described in section 6.5 of the MCTP SMBus/I²C Transport Binding Specification (DSP0237). It indicates support for ASF in the SMBus getUID command (see [Section 9.2.2.1.3.2](#)). It responds to any SMBus command using the MCTP command code. This ensures that the bus owner knows the X710-TM4 supports MCTP.

Note: MCTP commands over SMBus are received from any master address and are answered to the sender. There is no capturing of the bus owner address from any specific command.

9.7.4.2 MCTP over SMBus Special Features

The X710-TM4 supports the following optional feature of MCTP when running over SMBus:

- Simplified MCTP mode.
- Fairness arbitration.



9.7.4.2.1 Simplified MCTP Mode

For some point-to-point implementations of MCTP, the assembly process is simplified. In this mode, the destination EID, source EID, packet sequence number, Tag Owner (TO) bit and message tag are ignored and the assembly is based only on the *SOM* and *EOM* bits. This bit is set according to the *Simplified MCTP* bit in the MCTP configuration word in the NVM.

Note: This mode is not compliant with the MCTP specification.

In this mode, a Set EID command is not needed to start operation.

This mode is relevant only for MCTP over SMBus traffic and when the Redirection Sideband Interface is set to 10b (MCTP over SMBus only - no pass through).

9.7.4.2.2 Fairness Arbitration

When sending MCTP messages over SMBus and when fairness arbitration is enabled (see [Section 6.2.39.3](#)), the X710-TM4 should adhere to the fairness arbitration as defined in section 5.13 of DSP0237 when sending MCTP messages.

9.7.5 NC-SI over MCTP

The X710-TM4 support for NC-SI over MCTP is similar to the support for NC-SI over RBT with the following exceptions:

1. A set of new NC-SI OEM commands used to expose the NC-SI over MCTP capabilities.
2. The format of the packets is modified to account for the new transport layer as described in the sections that follow.



9.7.5.1 NC-SI Packets Format

NC-SI over MCTP defines two different message type for pass through and for control packets.

Packets with a message type equal to the *Control packets message type* field (default = 0x02) in the NVM are NC-SI control packets (commands, responses and AENs) and packets with a message type equal to the *Pass through packets message type* field (default = 0x03) in the NVM are NC-SI pass through packets

9.7.5.1.1 Control Packets

The format used for control packets (commands, responses and AENs) is as follows:

Table 9-77. NC-SI Over MCTP Over PCIe/SMBus Message Format

+0								+1								+2								+3									
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0		
SMBus or PCIe header																																	
MCTP Reserved				Header version = 1				Destination endpoint ID								Source endpoint ID								S O M		E O M		SEQ#		T O = 1		Tag	
I C = 0		Message Type = Control Packets Message type (0x02)						MC ID = 0x00								Header revision								Reserved									
IID								Command								Channel ID ¹								Reserved				Payload Length[11:8]					
Payload Length[7:0]								Reserved																									
Reserved								Reserved																									
Reserved								Command Data																									
....																																	
Command Data																Checksum																	
Checksum																																	

1. The channel ID is defined as described in Section 8.2.2.2.



Note that the MAC header and MAC FCS present when working over NC-SI are not part of the packet in MCTP mode.



9.7.5.1.2 Pass Through Packets

The format used for Pass Through packets are as follows. This format is the same for either packets received from the network or packets received from the host.

The CRC is never included in the packet. In receive, the CRC is checked and removed by the X710-TM4 in transmit, the CRC is added by the X710-TM4.

Table 9-78. Ethernet Over MCTP Over PCIe/SMBus Message Format

+0								+1								+2								+3									
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0		
SMBus or PCIe header																																	
MCTP Reserved				Header version = 1				Destination endpoint ID								Source endpoint ID								S O M		E O M		SEQ#		T O = 1		Tag	
I C = 0		Message Type = Pass Through Packets Control Type						DA																									
DA																SA																	
SA																																	
SA								Ether type																Ethernet Packet									
Ethernet packet																																	
....																																	
....																																	

9.7.6 PLDM over MCTP

Enabling PLDM over MCTP for control traffic is done by setting the *Control Interface* field in the *Common Manageability Parameters* NVM word to MCTP over SMBus/PCI and by setting the *PLDM* bit in *Common Manageability Parameters 2* NVM word.

9.7.7 OEM Commands

Enabling OEM commands over MCTP for control traffic is done by setting the *Control Interface* field in the *Common Manageability Parameters* NVM word to MCTP over SMBus/PCI and by setting the *OEM commands* bit in *Common Manageability Parameters 2* NVM word.



9.7.8 MCTP Programming

The MCTP programming model is based on:

1. A set of MCTP commands used for the discovery process and for the link management. The list of supported commands is described in section [Section 9.7.8.1](#).
2. A subset of the NC-SI commands used in the regular NC-SI interface, including all the OEM commands as described in [Section 9.6.2](#) (NC-SI programming I/F). The specific commands supported are listed in [Table 9-28](#) and [Table 9-31](#).

Note: For all MCTP commands (both native MCTP commands and NCSI over MCTP), the response uses the Msg tag received in the request with *TO* bit cleared.

9.7.8.1 MCTP Commands Support

[Table 9-79](#) lists the MCTP commands supported by the X710-TM4.

Table 9-79. MCTP Commands Support

Command Code	Command Name	General Description	X710-TM4 Support As Initiator	X710-TM4 Support As Responder
0x00	Reserved	Reserved	–	–
0x01	Set Endpoint ID	Assigns an EID to the endpoint at the given physical address.	N/A	Yes
0x02	Get Endpoint ID	Returns the EID presently assigned to an endpoint. Also returns information about what type the endpoint is and its level of use of static EIDs. See Section 9.7.8.1.1 for details.	No	Yes
0x03	Get Endpoint UUID	Retrieves a per-device unique UUID associated with the endpoint. See Section 9.7.8.1.2 for details.	No	Yes
0x04	Get MCTP Version Support	Lists which versions of the MCTP control protocol are supported on an endpoint. See Section 9.7.8.1.3 for details.	No	Yes
0x05	Get Message Type Support	Lists the message types that an endpoint supports. See Section 9.7.8.1.4 for details.	No	Yes
0x06	Get Vendor Defined Message Support	Used to discover an MCTP endpoint’s vendor specific MCTP extensions and capabilities. See Section 9.7.8.1.5 for details.	No	Yes
0x07	Resolve Endpoint ID	Used to get the physical address associated with a given EID.	Yes	N/A
0x08	Allocate Endpoint IDs	Used by the bus owner to allocate a pool of EIDs to an MCTP bridge.	N/A	N/A
0x09	Routing Information Update	Used by the bus owner to extend or update the routing information that is maintained by an MCTP bridge.	N/A	N/A

**Table 9-79. MCTP Commands Support (Continued)**

Command Code	Command Name	General Description	X710-TM4 Support As Initiator	X710-TM4 Support As Responder
0x0A	Get Routing Table Entries	Used to request an MCTP bridge to return data corresponding to its present routing table entries.	No	N/A
0x0B	Prepare for Endpoint Discovery	Used to direct endpoints to clear their discovered flags to enable them to respond to the Endpoint Discovery command.	N/A	Yes ¹
0x0C	Endpoint Discovery	Used to discover MCTP-capable devices on a bus, provided that another discovery mechanism is not defined for the particular physical medium.	No	Yes ¹
0x0D	Discovery Notify	Used to notify the bus owner that an MCTP device has become available on the bus.	Yes ¹	N/A
0x0E	Get Network ID	Used to get the MCTP network ID.	No	No
0x0F	Query Hop	Used to discover what bridges, if any, are in the path to a given target endpoint and what transmission unit sizes the bridges will pass for a given message type when routing to the target endpoint.	No	No

1. These commands are supported only for MCTP over PCIe.

9.7.8.1.1 Get Endpoint ID

The get endpoint ID response of the X710-TM4 is listed in the following table:

Byte	Description	Value
1	Completion Code	
2	Endpoint ID	0x00: EID not yet assigned. Otherwise - returns EID assigned using Set Endpoint ID command.
3	Endpoint Type	0x00 = Dynamic EID, simple endpoint.
4	Medium Specific	SMBus - 0x01 = Fairness arbitration protocol supported. PCIe: 0x00.

9.7.8.1.2 Get Endpoint UUID

The UUID returned is calculated according to the following function:

- Time Low = Read from MCTP UUID - Time Low LSB/MSB NVM words of Sideband Configuration Structure.
- Time mid = Read from MCTP UUID - Time Mid NVM word of Sideband Configuration Structure
- Time High and version = Read from MCTP UUID - Time High and version NVM word of Sideband Configuration Structure
- Clock Sec and Reserved = Read from MCTP UUID - Clock Seq NVM word of Sideband Configuration Structure
- Node = MAC address as taken from the *GLPCI_SERL* and *GLPCI_SERH* registers.



9.7.8.1.3 Get MCTP Version Support

The following table lists the returned value according to the requested message type. The list of supported message types is based on the protocols enabled in the NVM and should be the same as the list reported in the Get message type support command.

Byte	Description	Message Type						
		0xFF (Base)	0x00 (Control Protocol Message)	0x01 (PLDM)	0x02 (NC-SI Over MCTP) ¹	0x03 (Ethernet) ²	0x7E (PCIe Based VDM Messages) ³	All Other Or Unsupported Messages
1	Completion Code	0x0						0x80
2	Version Number Entry Count	3	3	1	2	2	2	0
6:3	Version Number Entry	0xF1F0FF00 (1.0)	0xF1F0FF00 (1.0)	0xF1F0F000 (1.0.0)	0xF1F0F000 (1.0.0)	0xF1F0F000 (1.0.0)	0xF1F0FF00 (1.0)	0
10:7	Version Number Entry 2	0xF1F1F000 (1.1.0)	0xF1F1F000 (1.1.0)		0xF1F1F000 (1.1.0)	0xF1F1F000 (1.1.0)	0xF1F1F000 (1.1.0)	
14:11	Version Number Entry 3	0xF1F2F000 (1.2.0)	0xF1F2F000 (1.2.0)					

1. If NC-SI is supported in the current configuration over this medium.
2. If NC-SI pass-through is supported in the current configuration over this medium.
3. If OEM messages are supported in the current configuration over this medium.

9.7.8.1.4 Get Message Type Support Command

The get message type support response of the X710-TM4 is listed in the following table:

Byte	Description	Value
1	Completion Code	0x00.
2	MCTP Message Type Count	0x01/0x02/0x03 - The X710-TM4 supports up to four additional message types, depending on the mode of operation and the bus used.
3:5	List of Message Type Numbers	0x01 (PLDM over MCTP) - If PLDM is supported.
		0x02 (NC-SI over MCTP) - If NC-SI is supported.
		0x03 (Ethernet). If NC-SI and pass through are supported.
		0x7E (PCIe based VDM messages) - over PCIe. Over SMBus also if OEM commands are supported.



9.7.8.1.5 Get Vendor Defined Message Support Command

The get vendor defined message type support response of X710-TM4 is listed in the following table if vendor ID set selector equals 0x00:

Byte	Description	Value
1	Completion Code	0x00.
2	Vendor ID Set Selector	0xFF = No more capability sets.
3:5	Vendor ID	0x008086 (PCI ID indicator + Intel vendor ID).
6:7	Version	0x0100 (version 1.0).

9.7.8.1.6 Set Endpoint ID Command

The X710-TM4 supports the set EID and force EID operations defined in the Set Endpoint ID command. When operating over PCIe, the set discovered flag operation is also supported. As endpoints in the X710-TM4 can be set only through their own interface, set EID and force EID are equivalent. The Reset EID operation is not supported by the X710-TM4.



The Set Endpoint ID response of the X710-TM4 is listed in the following table:

Byte	Description	Value
1	Completion Code	0x00.
2	Completion Status	[7:6] = 00b - Reserved.
		[5:4] = 00b - EID assignment accepted.
		[3:2] = 00b - Reserved.
		[1:0] = 00b - Device does not use an EID pool.
3	EID Setting	If the EID setting was accepted, this value matches the EID passed in the request. Otherwise, this value returns the present EID setting.
4	EID Pool Size	Always return a zero.

9.8 Host Isolate Support

If a MC decides that malicious software prevents its usage of the LAN, it might decide to isolate the NIC from its driver. This is done using the TCO reset command ([Section 9.6.4.12](#)).

If TCO isolate is enabled in the NVM ([Section 6.2.36.4](#)), The TCO Isolate command disables PCIe write operations to the LAN port. As the software device driver needs to access the CSR space in order to provide descriptors to the NIC, this operation also stops the network traffic including OS2BMC and MC-to-OS traffic as soon as the existing transmit and receive descriptor queues are exhausted.

MCTP over PCIe VDM are still available in this mode.



10.0 Programming Interface

10.1 Introduction

This section details the programmer visible state inside the X710-TM4. In some cases, it describes hardware structures invisible to software in order to clarify a concept.

The X710-TM4 address space is mapped into four regions with PCI Base Address registers described in [Section 11.2.6.1](#). These regions are listed in the following table.

Table 10-1. Address Space Regions

Addressable Content	How Mapped	Size of Region
Memory BAR (internal registers, memories and Flash)	Direct memory-mapped	4 MB - 16 MB
I/O BAR (optional internal registers)	I/O window mapped	32 bytes ¹
MSI-X BAR (optional)	Direct memory-mapped	32 KB
Expansion ROM BAR (optional)	Direct memory-mapped	64 KB - 8 MB

1. The internal registers can be accessed through I/O space indirectly as explained in the sections that follow.

Rules for unsupported accesses:

- Accesses to non-implemented or disabled regions within a BAR are dropped for write accesses or responded with arbitrary data for read accesses. A PCIe error event is not generated and completions return with successful status.
- [Section 3.1.2.2](#) describes supported PCIe access sizes to each of the BARs and its components.

10.1.1 Access Mechanisms

10.1.1.1 Memory-mapped Access to Internal Registers and Memories

The internal registers and memories might be accessed as direct memory-mapped offsets from the Base Address register (BAR0 or BAR 0/1 see [Section 11.2.6.1](#)). See [Section 10.2](#) for the appropriate offset for each specific internal register.

In IOV mode, this area is partially duplicated per VF. All replications contain only the subset of the register set that is available for VF programming. See [Section 10.4](#) for the appropriate offset for each specific internal register for VFs.



10.1.1.2 Memory-mapped Access to Flash

The external Flash can be accessed using direct memory-mapped offsets from the memory base address register (BAR0 in 32-bit addressing or BAR0/BAR1 in 64-bit addressing see [Section 11.2.6.1](#)). See [Table 10-4](#) for the location of Flash memory within the memory BAR. Access to Flash memory is restricted to the first 64 KB when GLPCI_LBARCTRL.FLASH_EXPOSE is set to 0b.

See [Section 3.4](#) for details on accessing the NVM.

10.1.1.3 Memory-mapped Access to MSI-X Tables

The MSI-X tables can be accessed as direct memory-mapped offsets from the base address register (see BAR3 or BAR3/4 [Section 11.2.6.1](#)). See [Section 11.3.3](#) for the appropriate offset for each specific internal MSI-X register.

In IOV mode, this area is duplicated per VF. It requires a memory space of the maximum between 16 KB and the page size.

10.1.1.4 Memory-mapped Access to Expansion ROM

The external Flash can also be accessed as a memory-mapped expansion ROM. Accesses to offsets starting from the expansion ROM base address (see [Section 11.2.6.2](#)) reference the Flash provided that access is enabled from NVM, and if the expansion ROM base address register contains a valid (non-zero) base memory address.

10.1.1.5 I/O-mapped Access to Internal Registers

To support pre-boot operation, all internal registers in the regular CSR space can be accessed using I/O operations. I/O accesses are supported only if an I/O base address is allocated and mapped (see [BAR2 Section 11.2.6.1](#)). I/O address decoding is enabled in the PCIe configuration.

When an I/O BAR is mapped, the I/O address range allocated opens a 32-byte window in the system I/O address map. Within this window, two I/O addressable registers are implemented: IOADDR and IODATA. The IOADDR register is used to specify a reference to an internal register and the IODATA register is used as a window to the register address specified by IOADDR as listed in [Table 10-2](#).

Table 10-2. IOADDR and IODATA in I/O Address Space

Offset	Abbreviation	Name	RW	Size
0x00	IOADDR	Internal Register Address. Covers the (4 MB - 64 KB) CSR space. 0x00000-0x3EFFFF – Internal registers. 0x00000-0x7FFFFFF – Internal registers. 0x3F0000-0xFFFFFFFF – Undefined.	RW	4 bytes
0x04	IODATA	Data field for reads or writes to the internal register location as identified by the current value in IOADDR. All 32 bits of this register can be read from and written to.	RW	4 bytes
0x08 – 0x1F	Reserved	Reserved.	RO	4 bytes



10.1.1.5.1 IOADDR (I/O Offset 0x00)

The IOADDR register must always be written as a Dword access. [Section 3.1.2.3](#) describes how other access sizes are handled.

Note: For software programmers, the IN and OUT instructions must be used to cause I/O cycles to be used on the PCIe bus. Because writes must be to a 32-bit quantity, the source register of the OUT instruction must be EAX (the only 32-bit register supported by the OUT command). For reads, the IN instruction can have any size target register, but it is recommended that the 32-bit EAX register be used.

At hardware reset (LAN_PWR_GOOD) or PCI reset, this register value resets to 0x00000000. Once written, the value is retained until the next write or reset.

10.1.1.5.2 IODATA (I/O Offset 0x04)

The IODATA register must always be written as a Dword access (assuming the IOADDR register contains a value for the internal register space). Reads to IODATA returns a Dword of data. [Section 3.1.2.3](#) describes how other access sizes are handled.

Note: For software programmers, the IN and OUT instructions must be used to cause I/O cycles to be used on the PCIe bus. Where 32-bit quantities are required on writes, the source register of the OUT instruction must be EAX (the only 32-bit register supported by the OUT command).

Note: There are no special software timing requirements on accesses to IOADDR or IODATA. All accesses are immediate, except when data is not readily available or acceptable. In this case, the X710-TM4 delays the results through normal bus methods (for example, split transaction or transaction retry).

Because a register read or write takes two I/O cycles to complete, software must provide a guarantee that the two I/O cycles occur as an atomic operation. Otherwise, results can be non-deterministic from the software viewpoint.

10.1.1.5.3 Undefined I/O Offsets

I/O offsets 0x08 through 0x1F are considered to be reserved offsets with the I/O window. Write accesses to these addresses are discarded. Read accesses to these addresses return undefined content.

10.1.1.6 Configuration Access to Internal Registers

To support legacy pre-boot 16-bit operating environments without requiring I/O address space, the X710-TM4 enables accessing CSRs via configuration address space by mapping the IOADDR and IODATA registers into configuration address space. If the GLPCI_CAPSUP.CSR_CONF_EN bit is set to 1b, access to CSRs via configuration address space is enabled. The register mappings in this case are listed in [Table 10-3](#).

**Table 10-3. IOADDR and IODATA in Configuration Address Space**

Configuration Address	Abbreviation	Name	RW	Size
0x98	IOADDR	Internal Register Address. Covers the (4 MB - 64 KB) CSR space. 0x00000-0x3FFFFFF - Internal registers. 0x3F0000-0xFFFFFFFF - Undefined.	RW	4 bytes
0x9C	IODATA	Data field for reads or writes to the internal register location as identified by the current value in IOADDR. All 32 bits of this register can be read from and written to.	RW	4 bytes

Software writes data to an internal CSR via configuration space in the following manner:

1. CSR address is written to IOADDR where:
 - a. Bit 31 (*IOADDR.Configuration IO Access Enable*) of IOADDR should be set to 1b.
 - b. Bits 30:0 of IOADDR should hold the actual address of the internal register being written to.
2. Data to be written is written into IODATA.
 - IODATA is used as a window to the register address specified by IOADDR. As a result, the data written to IODATA is written into the CSR pointed to by bits 30:0 of IOADDR.
3. IOADDR is cleared (all bits [31:0]), to avoid un-intentional CSR read operations (that might cause clear by read) by other applications scanning the configuration space.

Software reads data from an internal CSR via configuration space in the following manner:

1. The CSR address is written to IOADDR where:
 - a. Bit 31 (*IOADDR.Configuration IO Access Enable*) of IOADDR should be set to 1b.
 - b. Bits 30:0 of IOADDR should hold the actual address of the internal register being read.
2. The CSR value is read from IODATA.
 - a. IODATA is used as a window to the register address specified by IOADDR. As a result, the data read from IODATA is the data of the CSR pointed to by bits 30:0 of IOADDR.
3. IOADDR is cleared (all bits [31:0]), to avoid un-intentional CSR read operations (that might cause clear by read) by other applications scanning the configuration space.

Note: In the event that the `GLPCI_CAPSUP.CSR_CONF_EN` bit is cleared, accesses to IOADDR and IODATA via the configuration address space are ignored and have no effect on the register and the CSRs referenced by IOADDR.

When a function is in D3 state, software should not attempt to access CSRs via IOADDR and IODATA.

To enable CSR access via configuration space, software should set bit 31 (*IOADDR.Configuration IO Access Enable*) of IOADDR to 1b. Software should clear bit 31 of IOADDR after completing CSR access to avoid an unintentional clear by read operation, by another application scanning the configuration address space. Software should also clear bits 30:0 of IOADDR to remove any trace of previous accesses to the configuration space (see previous flows).

Bit 31 of IOADDR (*IOADDR.Configuration IO Access Enable*) has no effect when initiating access via IO address space.



10.1.2 Memory BAR

10.1.2.1 PF BAR Structure

The memory BAR provides access to internal CSRs and to the external Flash (NVM). This section describes where each of these is located within the BAR space.

The following configuration parameters define the structure of the PF memory BAR 0:

- *Flash_Expose* bit from the NVM (or the GLPCI_LBARCTRL.FLASH_EXPOSE CSR bit)
 - 0b = Flash memory is not mapped in the memory BAR
 - 1b = Flash memory is mapped in the memory BAR. Hardware default; Flash memory is exposed when during initialization the Flash is found to be blank or in error.
- *Flash Size* field from the NVM (or the GLPCI_LBARCTRL.FL_SIZE CSR field) - size is calculated as 64 KB * (2 ** Flash Size). Default value is 8 MB

The table that follows lists all supported partitions of the memory BAR as a function of the previously described parameters. Other combinations of the parameters (not covered in the table) are not supported and considered reserved for future expansion. The following rules apply:

- CSR space is located from the beginning of the BAR until address (4 MB-64 KB-1) (such as the first 4 MB - 64 KB)
- CSR space is located from the beginning of the BAR until address (8 MB-64 KB-1).
- The Flash space (if exposed) is always aligned to multiples of its size.

The default configuration of the memory BAR (like when the Flash is empty) is defined by hardware default values of the read-only GLPCI_LBARCTRL CSR. GLPCI_LBARCTRL is loaded from the NVM during normal operation (such as when Flash contents are valid).

Table 10-4. Structure of the PF Memory BAR

Flash Expose	Flash Size	Flash Space		BAR Size
		Min Addr	Max Addr	
0	x	x	x	4 MB
		x	x	4 MB
		x	x	8 MB
1	2 MB	4 MB	6 MB	8 MB
		4 MB	6 MB	8 MB
		8 MB	10 MB	16 MB
	4 MB	4 MB	8 MB	8 MB
		4 MB	8 MB	8 MB
		8 MB	12 MB	16 MB
	8 MB	8 MB	16 MB	16 MB
		8 MB	16 MB	16 MB
		8 MB	16 MB	16 MB



10.1.2.2 VF BAR Structure

The VF memory BAR provides access to on-die CSRs for VFs.

The size of the VF BAR is Max (64K, Page size).

10.1.3 MSI-X BAR

The structure of the MSI-X BAR is described in [Section 11.3.3](#) (for a PF) and [Section 11.5.3.1](#) (for a VF).

10.1.4 CSR Organization and Mapping

This section describes how CSRs are mapped into the PF and VF memory BAR. This section does not apply to the following address space:

- The MSI-X BAR (defined per the PCI specifications)

10.1.4.1 Mapping by Scope

Registers are associated with a scope. A scope is a set of attributes for a register that define which functions can access the register and how many instances exist for the register. The following table lists the different scopes.

Table 10-5. Scope Mapping

Scope	PF/VF	Quantity	Exposure	Comments
GL	PF	1	To all PFs.	
GLVF	PF, VF	1	To all PFs and VFs.	Registers are RO.
PRT	PF	4	Each PF has access to the registers of the port it is associated with.	Registers are shared by all PFs on a port.
PRTVF	PF, VF	4	Each PF or VF has access to the registers of the port it is associated with (for a VF, the port is the port the PF is associated with).	Registers are RO. Registers are shared by all PFs and VFs on a port.
PF	PF	16	Each PF has access to its copy only.	
VF	PF, VF	128	Each PF has access to the registers of its VFs only. Each VF has access to its copy only.	
VP	PF	128	Each PF has access to the registers of its VFs only.	These registers control VF functionality.

**Table 10-5. Scope Mapping**

Scope	PF/VF	Quantity	Exposure	Comments
Q	PF, VF	1536	Each PF has access to the registers allocated to it (including its VFs). Each VF has access to the registers allocated to it.	
VSI	PF	384	All PFs.	Registers are shared by all PFs.
INTPF	PF	512	Each PF has access to the registers allocated to it. ¹	
INTVF	PF, VF	512	Each VF has access to the registers allocated to it. ¹ Each PF has access to the registers of its VFs only. ¹	
INTVP	PF	512	Each PF has access to the registers of its VFs only. ¹	These registers handle VF interrupts.

1. Note that the register descriptions in section Device Registers - PF and section Device Registers - VF list 512 instances per register. However, the actual number of registers exposed to a function is listed in [Table 7-127](#).

10.1.5 Register Conventions

All registers in the X710-TM4 are defined to be 32 bits and accessed as 32-bit Dwords. There are some exceptions to this rule:

- Register pairs where two 32-bit registers make up a larger 64-bit logical unit.
- Accesses to Flash memory via expansion ROM space, secondary BAR space or the I/O space) might be byte, word or double word accesses. I/O accesses are limited to Dword accesses (see [Section 10.1.1.5](#)).
- Accesses to BAR0 of a VF might be byte, word or Dword accesses. 64-bit (Qword) accesses to this BAR are completed with an Completer Abort (CA) error.
- Access to the MSI-X BAR of the PFs and the VFs might be Dword or Qword accesses.

Reserved bit positions — Some registers contain certain bits that are marked as reserved. Writes to a reserved field must set the field to its initial value unless specified differently in the field description. Reads from registers containing reserved bits might return indeterminate values in the reserved bit-positions unless read values are explicitly stated. When read, these reserved bits should be ignored by software.

Reserved and/or undefined addresses — Any register address not explicitly declared in this document should be considered to be reserved, and should not be written to. Writing to reserved or undefined register addresses might cause indeterminate behavior. Reads from reserved or undefined configuration register addresses might return indeterminate values unless read values are explicitly stated for specific addresses.

Initial values — Most registers define the initial hardware values prior to being programmed. In some cases, hardware initial values are undefined and is listed as such via the text undefined, unknown, or X. Such configuration values might need to be set via NVM configuration or via software in order for proper operation to occur; this need is dependent on the function of the bit. Other registers might cite a hardware default which is overridden by a higher-precedence operation. Operations that might supersede hardware defaults might include a valid NVM load, completion of a hardware operation (such as hardware auto-negotiation), or writing of a different register whose value is then reflected in another bit.\



For registers that should be accessed as 32-bit Dwords, partial writes (less than a 32-bit Dword) does not take effect (the write is ignored). Partial reads returns all 32 bits of data regardless of the byte enables.

Note: Partial reads to clear-on-read registers (ICR) can have unexpected results since all 32 bits are actually read regardless of the byte enables. Partial reads should not be done.

All statistics registers are implemented as 32-bit registers. Though some logical statistics registers represent counters in excess of 32 bits in width, registers must be accessed using 32-bit operations (for example, independent access to each 32-bit field). When reading 64-bit statistics registers the least significant 32-bit register should be read first.

See special notes for VLAN filter table, multicast table arrays and packet buffer memory that appear in the specific register definitions.

10.1.5.1 Register Abbreviation Naming Conventions

The register abbreviation naming follows (in most cases) the following rules:

- Abbreviation starts with the register’s scoping. It could be port registers (PRT), PF registers (PF) and so on. See [Table 10-5](#) for the complete list of register’s scopes.
- Then it follows by the register’s main block like DCB (for DCB registers), QF (for queue filters) and so on.
- Then it follows by a few characters that summarize the register’s name.

10.1.6 Register’s Fields Attribute

The following table lists the access type of registers' bit fields. The access rights of the PFs and the VFs to the entire registers are defined per PF and VF registers. In some cases, the PFs and VFs might have Read Only (RO) access rights to registers that can be programmed by the internal logic (either auto-load from the NVM or programmed by the firmware). Registers defined as RO access, override any access type defined for its fields.

Abbreviation	Description
RO	Read Only - A register bit field with this attribute can be read. Writes have no effect on the bit field value. Used also for bits of RO/V type.
RO/V	Read Only Variant - A register bit field with this attribute can be read. Writes have no effect on the bit field value. The hardware may change the value returned. Software should not expect a static value.
ROCV	Read Only Clear Value - A register bit field with this attribute can be read. The value returned on the read might be different in each read (typically a counter) and the value is cleared after the read. A write has no effect.
RSV	Reserved - A register bit field with this attribute can be read and returns an indeterministic value. Writes must set the bit field to its initial value unless specified differently in the field description.
RW	Read/Write - A register with this attribute can be read and written. Read return the default value, the last value written, or updated status from a previous operation. The field description specifies the field's actual behavior.
RW/V	Read-Write, Variant, hardware loadable - Software can read from and write to field. Hardware may modify field value. Hardware write have higher priority compare to software write.



Abbreviation	Description
RCW	Read Clear / Write - A register bit field with this attribute can be written or read. The value returned on the read might be different than the value written (typically a counter) and the value is cleared after the read.
RW1C	Read/Write 1 to Clear - A register bit field with this attribute can be read and written. Writing an individual bit within the field to a 1b clears (sets to 0b) the corresponding bit and a write of a 0b has no effect. The value read might return the last value written or the status of a previous operation. The field description specifies the field's actual read behavior.
RW1S	Read/Write 1 to Set - A register bit field with this attribute can be read and written. Writing an individual bit within the field to a 1b sets (sets to 1b) the corresponding bit and a write of a 0b has no effect. The value read might return the last value written or the status of a previous operation. The field description specifies the field's actual read behavior.
RWC	Read/Write Clear - A register bit field with this attribute can be written or read. The value returned on the read might be different than the value written (typically a counter) and the value is cleared after a write of any value.
SC	Self Clear - A register bit field with this attribute can be written or read. The value returned on the read might be different than the value written (typically an action request), as hardware clears the bit when the action is done/activated.
WO	Write Only - A register bit field with this attribute can be only written. A read returns a fixed value not reflecting the content of the register. This is used for keys that should not be visible to other software agents.



NOTE: *This page intentionally left blank.*



10.2 Device Registers - PF

10.2.1 BAR0 Registers Summary

Table 10-7. BAR0 Registers Summary

Offset / Alias Offset	Abbreviation	Name	Section
PF - General Registers			
0x000B612C	GLGEN_STAT	Global Status	Section 10.2.1.1.1
0x000B8120	PRTGEN_CNF	General Port Configuration	Section 10.2.1.1.2
0x000B8160	PRTGEN_CNF2	General Port Configuration2	Section 10.2.1.1.3
0x000B8100	PRTGEN_STATUS	General Port Status	Section 10.2.1.1.4
0x000B8188	GLGEN_RSTAT	Global Reset Status	Section 10.2.1.1.5
0x000B8190	GLGEN_RTRIG	Global Reset Trigger	Section 10.2.1.1.6
0x000B8180	GLGEN_RSTCTL	Global Reset Delay	Section 10.2.1.1.7
0x00090000 + 0x4*VSI, VSI=0...383	VSIGEN_RTRIG[VSI]	VM Reset Trigger	Section 10.2.1.1.8
0x00090800 + 0x4*VSI, VSI=0...383	VSIGEN_RSTAT[VSI]	VM Reset Status	Section 10.2.1.1.9
0x00091C00 + 0x4*VF, VF=0...127	VPGEN_VFRSTAT[VF]	VF Reset Status	Section 10.2.1.1.10
0x00074400 + 0x4*VF, VF=0...127	VFGEN_RSTAT[VF]	VF Reset Status	Section 10.2.1.1.11
0x00091800 + 0x4*VF, VF=0...127	VPGEN_VFRTRIG[VF]	VF Reset Trigger	Section 10.2.1.1.12
0x00092600 + 0x4*n, n=0...3	GLGEN_VFLRSTAT[n]	Global VF Level Reset Status	Section 10.2.1.1.13
0x000B8184	GLGEN_CLKSTAT	Global Clock Status	Section 10.2.1.1.14
0x00088100 + 0x4*n, n=0...29	GLGEN_GPIO_CTL[n]	Global GPIO Control	Section 10.2.1.1.15
0x00088178	GLGEN_LED_CTL	Global LED Control	Section 10.2.1.1.16
0x0008817C	GLGEN_GPIO_STAT	Global GPIO Status	Section 10.2.1.1.17
0x00088180	GLGEN_GPIO_TRANSIT	Global GPIO Transition Status	Section 10.2.1.1.18
0x00088184	GLGEN_GPIO_SET	Global GPIO Set	Section 10.2.1.1.19
0x000881C0 + 0x4*n, n=0...3	GLGEN_MDIO_I2C_SEL[n]	Global MDIO or I ² C Select	Section 10.2.1.1.20
0x000881D0 + 0x4*n, n=0...3	GLGEN_MDIO_CTRL[n]	MDIO Control	Section 10.2.1.1.21
0x0008818C + 0x4*n, n=0...3	GLGEN_MSCA[n]	MDI Single Command and Address	Section 10.2.1.1.22
0x0008819C + 0x4*n, n=0...3	GLGEN_MSRWD[n]	MDI Single Read and Write Data	Section 10.2.1.1.23
0x000881E0 + 0x4*n, n=0...3	GLGEN_I2CCMD[n]	I ² C Command	Section 10.2.1.1.24
0x000881AC + 0x4*n, n=0...3	GLGEN_I2CPARAMS[n]	I ² C Parameters	Section 10.2.1.1.25
0x000881BC	GLVFGEN_TIMER	Global Device Timer	Section 10.2.1.1.26
0x00092400	PFGEN_CTRL	PF Control	Section 10.2.1.1.27
0x00088000	PFGEN_STATE	PF State	Section 10.2.1.1.28



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00092500	PFGEN_DRUN	PF Driver Unload	Section 10.2.1.1.29
0x001C0480	PFGEN_PORTNUM	LAN Port Number	Section 10.2.1.1.30
0x00083048	GL_FWSTS	Firmware Status Register	Section 10.2.1.1.31
PF - PCIe Registers			
0x000BE100	PFPCI_SUBSYSID	PF PCIe Subsystem ID	Section 10.2.1.2.3
0x000BE300	PFPCI_PM	PCIe PM	Section 10.2.1.2.1
0x000BE518	GLPCI_VENDORID	PCIe Vendor ID	Section 10.2.1.2.2
0x001C0AB4	GLGEN_PCIFCNCNT	PCI Function Count	Section 10.2.1.2.4
0x000BE494	GLPCI_CNF2	PCIe* Global Config 2	Section 10.2.1.2.5
0x000BE484	GLPCI_LBARCTRL	PCI BAR Control	Section 10.2.1.2.6
0x000BE4C0	GLPCI_CNF	PCIe* Global Config	Section 10.2.1.2.7
0x000BE4A8	GLPCI_CAPSUP	PCIe Capabilities Support	Section 10.2.1.2.8
0x000BE4AC	GLPCI_LINKCAP	PCIe Link Capabilities	Section 10.2.1.2.9
0x000BE4A4	GLPCI_CAPCTRL	PCIe Capabilities Control	Section 10.2.1.2.10
0x000BE480	GLTPH_CTRL	TPH Control Register	Section 10.2.1.2.11
0x000BE498	GLPCI_SERL	PCIe Serial Number MAC Address Low	Section 10.2.1.2.12
0x000BE49C	GLPCI_SERH	PCIe Serial Number MAC Address High	Section 10.2.1.2.13
0x000BE400	PFPCI_CLASS	PCIe Storage Class	Section 10.2.1.2.14
0x000BE000	PFPCI_CNF	PCIe PF Configuration	Section 10.2.1.2.15
0x000BE200	PFPCI_FUNC	PCIe Functions Configuration	Section 10.2.1.2.16
0x000BE180	PFPCI_FUNC2	PCIe Functions Configuration 2	Section 10.2.1.2.17
0x000BE4B8	GLPCI_VFSUP	PCIe VF Capabilities Support	Section 10.2.1.2.18
0x0009C480	GLPCI_DREVID	PCIe Default Revision ID	Section 10.2.1.2.19
0x0009C180	PFPCI_FACTPS	Function Active and Power State	Section 10.2.1.2.20
0x000BE280	PFPCI_STATUS1	PCIe Function Status 1	Section 10.2.1.2.21
0x0009C000	PF_FUNC_RID	Function Requester ID Information Register	Section 10.2.1.2.22
0x000BE4B0	GLPCI_PMSUP	PCIe PM Support	Section 10.2.1.2.23
0x000BE490	GLPCI_PWRDATA	PCIe Power Data Register	Section 10.2.1.2.24
0x000BE080	PFPCI_DEVID	PCIe PF Device ID	Section 10.2.1.2.25
0x000BE4B4	GLPCI_REVID	PCIe Revision ID	Section 10.2.1.2.26
0x000BE48C	GLPCI_SUBVENID	PCIe Subsystem ID	Section 10.2.1.2.27
0x0009C48C	GLPCI_GSCL_1	PCIe* Statistic Control Register #1	Section 10.2.1.2.28
0x0009C490	GLPCI_GSCL_2	PCIe* Statistic Control Registers #2	Section 10.2.1.2.29
0x0009C494 + 0x4*n, n=0...3	GLPCI_GSCL_5_8[n]	PCIe* Statistic Control Register #5...#8	Section 10.2.1.2.30
0x0009C4A4 + 0x4*n, n=0...3	GLPCI_GSCN_0_3[n]	PCIe* Statistic Counter Registers #0...#3	Section 10.2.1.2.31
0x0009C488	GLPCI_BYTCTL	PCIe Byte Counter Low	Section 10.2.1.2.32
0x0009C484	GLPCI_BYTCTH	PCIe Byte Counter High	Section 10.2.1.2.33
0x0009C4F0	GLPCI_PM_MUX_PFB	PCIe Mux Selector For PFB	Section 10.2.1.2.38



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x0009C4EC	GLPCI_PQ_MAX_USED_SPC	PCIe PQs Max Used Space	Section 10.2.1.2.37
0x0009C4F4	GLPCI_PM_MUX_NPQ	PCIe Mux Selector For NPQs	Section 10.2.1.2.39
0x0009C4BC	GLPCI_PKTCT	PCIe Packet Counter	Section 10.2.1.2.36
0x0009C4F8	GLPCI_SPARE_BITS_0	PCIe Regs Spare Bits 0	Section 10.2.1.2.35
0x0009C4FC	GLPCI_SPARE_BITS_1	PCIe Regs Spare Bits 1	Section 10.2.1.2.34
0x000BE4F8	GLPCI_UPADD	PCIe Upper Address	Section 10.2.1.2.40
0x0009C4C0	GLPCI_LCBADD	PCIe LCB Address Port	Section 10.2.1.2.41
0x0009C4C4	GLPCI_LCBDATA	PCIe LCB Data Port	Section 10.2.1.2.42
0x0009C080	PF_PCI_CIAA	PCIe Configuration Indirect Access Address	Section 10.2.1.2.43
0x0009C100	PF_PCI_CIAD	PCIe Configuration Indirect Access Data	Section 10.2.1.2.44
0x0009C300	PFPCI_VMINDEX	PCIe VM Pending Index	Section 10.2.1.2.45
0x0009C880	PFPCI_VM_FLUSH_DONE	PCIe VM Flush Done	Section 10.2.1.2.48
0x0009C600 + 0x4*VF, VF=0...127	PFPCI_VF_FLUSH_DONE [VF]	PCIe VF Flush Done	Section 10.2.1.2.46
0x0009C800	PFPCI_PF_FLUSH_DONE	PCIe PF Flush Done	Section 10.2.1.2.47
0x0009C380	PFPCI_VMPEND	PCIe VM Pending Status	Section 10.2.1.2.49
PF - MAC Registers			
0x001E3260	PRTMAC_HSEC_CTL_RX_ENABLE_GPP	HSEC Control Receive ENABLE_GPP	Section 10.2.1.3.3
0x001E3370 + 0x10*n, n=0...8	PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA[n]	HSEC Control Transmit PAUSE_QUANTA	Section 10.2.1.3.12
0x001E34C0	PRTMAC_HSEC_CTL_TX_SA_PART2	HSEC Control Transmit SA_GPP_PART2	Section 10.2.1.3.10
0x001E3140	PRTMAC_HSEC_CTL_RX_PAUSE_SA_PART1	HSEC Control Receive PAUSE_SA_PART1	Section 10.2.1.3.9
0x001E3120	PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2	HSEC Control Receive PAUSE_DA_UCAST_PART2	Section 10.2.1.3.1
0x001E30D0	PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE	HSEC Control Transmit PAUSE_ENABLE	Section 10.2.1.3.4
0x001E2120	PRTGL_SAL	Port MAC Address Low	Section 10.2.1.3.16
0x001E30E0	PRTMAC_HSEC_CTL_RX_ENABLE_GCP	HSEC Control Receive ENABLE_GCP	Section 10.2.1.3.11
0x0008C480	PRTMAC_PCS_XAUI_SWAP_A	PCS_XAUI_SWAP_A	Section 10.2.1.3.18
0x001E30C0	PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE	HSEC Control Receive PFC Enable	Section 10.2.1.3.2
0x001E3360	PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL	HSEC Control Receive FORWARD_CONTROL	Section 10.2.1.3.6



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x001E3110	PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1	HSEC CONTROL Receive PAUSE_DA_UCAST_PART1	Section 10.2.1.3.5
0x001E2140	PRTGL_SAH	Port MAC Address High	Section 10.2.1.3.15
0x001E3150	PRTMAC_HSEC_CTL_RX_PAUSE_SA_PART2	HSEC Control Receive PAUSE_SA_PART2	Section 10.2.1.3.8
0x001E3400 + 0x10*n, n=0...8	PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER[n]	HSEC Control Transmit PAUSE_REFRESH_TIMER	Section 10.2.1.3.13
0x0008C484	PRTMAC_PCS_XAUI_SWAP_B	PCS_XAUI_SWAP_B	Section 10.2.1.3.17
0x001E32E0	PRTMAC_HSEC_CTL_RX_ENABLE_PPP	HSEC Control Receive ENABLE_PPP	Section 10.2.1.3.14
0x001E34B0	PRTMAC_HSEC_CTL_TX_SA_PART1	HSEC Control Transmit SA_GPP_PART1	Section 10.2.1.3.7
PF - Power Management Registers			
0x000B8140	PRTPM_GC	General Control	Section 10.2.1.4.1
0x001E4380	PRTPM_EEEC	Energy Efficient Ethernet (EEE) Control	Section 10.2.1.4.3
0x001E4360	PRTPM_EEER	Energy Efficient Ethernet (EEE) Register	Section 10.2.1.4.2
0x001E4320	PRTPM_EEE_STAT	Energy Efficient Ethernet (EEE) STATUS	Section 10.2.1.4.4
0x001E4400	PRTPM_EEEFWD	EEE Tx Control	Section 10.2.1.4.5
0x001E43E0	PRTPM_EEETXC	EEE Tx Control	Section 10.2.1.4.6
0x001E43C0	PRTPM_TLPIC	EEE Tx LPI Count	Section 10.2.1.4.7
0x001E43A0	PRTPM_RLPIC	EEE Rx LPI Count	Section 10.2.1.4.8
PF - Wake-Up and Proxying Registers			
0x0006C800	GLPM_WUMC	WU on MNG Control	Section 10.2.1.5.1
0x001E44C0 + 0x20*n, n=0...3	PRTPM_SAH[n]	MAC Address High	Section 10.2.1.5.3
0x001E4440 + 0x20*n, n=0...3	PRTPM_SAL[n]	MAC Address Low	Section 10.2.1.5.2
0x0006C000	PRTPM_FHFHR	Flexible Host Filter Header Removal	Section 10.2.1.5.4
0x0006B200	PFPW_WUC	Wake Up Control Register	Section 10.2.1.5.6
0x000B8080	PFPW_APM	APM Control Register	Section 10.2.1.5.5
0x0006B400	PFPW_WUFC	Wake Up Filter Control Register	Section 10.2.1.5.7
0x0006B600	PFPW_WUS	Wake Up Status Register	Section 10.2.1.5.8
0x0006A000 + 0x80*n, n=0...7	PFPW_FHFT_LENGTH[n]	Flexible Host Filter Table Length	Section 10.2.1.6
PF - NVM Registers			
0x000B6104	GLNVM_FLASHID	Flash ID Register	Section 10.2.1.6.1
0x000B6100	GLNVM_GENS	Global NVM General Status Register	Section 10.2.1.6.2
0x000B6108	GLNVM_FLA	Flash Access Register	Section 10.2.1.6.3
0x000B6110	GLNVM_SRCTL	Shadow RAM Control Register	Section 10.2.1.6.4



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x000B6114	GLNVM_SRDATA	Shadow RAM Read/Write Data	Section 10.2.1.6.5
0x000B6008	GLNVM_ULD	Unit Load Status	Section 10.2.1.6.6
0x000B6010 + 0x4*n, n=0...59	GLNVM_PROTCSR[n]	Protected CSR List	Section 10.2.1.6.7
PF - Analyzer Registers			
0x001C0A70 + 0x4*n, n=0...7	GL_SWT_L2TAGCTRL[n]	L2 Tag Control	Section 10.2.1.7.1
0x001C0B20	PRT_L2TAGSEN	L2 Tag - Enable	Section 10.2.1.7.2
PF - Switch Registers			
0x0026CFB8 + 0x4*n, n=0...1	GL_SWR_DEF_ACT_EN[n]	Switching Table Default Action Enable Bitmap	Section 10.2.1.8.1
0x00270200 + 0x4*n, n=0...35	GL_SWR_DEF_ACT[n]	Switching Table Default Action	Section 10.2.1.8.2
0x00044000 + 0x20*n, n=0...7	PRT_TCTUPR[n]	Port - TC Transmit UP Replacement	Section 10.2.1.8.3
PF - VSI Context			
0x00042000 + 0x4*VSI, VSI=0...383	VSI_TAR[VSI]	VSI Tag Accept Register	Section 10.2.1.9.1
PF - Interrupt Registers			
0x0003F800	GLINT_CTL	Global Interrupt Control	Section 10.2.1.10.1
0x0003F100	PFGEN_PORTMDIO_NUM	LAN Port MDIO Number	Section 10.2.1.10.2
0x00038780	PFINT_ICR0	PF Interrupt Zero Cause	Section 10.2.1.10.3
0x00038800	PFINT_ICR0_ENA	PF Interrupt Zero Cause Enablement	Section 10.2.1.10.4
0x00038480	PFINT_DYN_CTL0	PF Interrupt Zero Dynamic Control	Section 10.2.1.10.5
0x00038400	PFINT_STAT_CTL0	PF Interrupt Zero Static Control	Section 10.2.1.10.6
0x00038500	PFINT_LNKLST0	PF Interrupt Zero Linked List	Section 10.2.1.10.7
0x00034800 + 0x4*INTPF, INTPF=0...511	PFINT_DYN_CTLN[INTPF]	PF Interrupt N Dynamic Control	Section 10.2.1.10.8
0x00035000 + 0x4*INTPF, INTPF=0...511	PFINT_LNKLSTN[INTPF]	PF Interrupt N Linked List	Section 10.2.1.10.9
0x00038000 + 0x80*n, n=0...2	PFINT_ITR0[n]	PF Interrupt Throttling for Interrupt Zero	Section 10.2.1.10.10
0x00030000 + 0x800*n + 0x4*INTPF, n=0...2, INTPF=0...511	PFINT_ITRN[n,INTPF]	PF Interrupt Throttling for Interrupt N	Section 10.2.1.10.11
0x00038580	PFINT_RATE0	PF Interrupt Zero Rate Limit	Section 10.2.1.10.12
0x00035800 + 0x4*INTPF, INTPF=0...511	PFINT_RATEN[INTPF]	PF Interrupt N Rate Limit	Section 10.2.1.10.13
0x0003A000 + 0x4*Q, Q=0...1535	QINT_RQCTL[Q]	Receive Queue Interrupt Cause Control	Section 10.2.1.10.14
0x0003C000 + 0x4*Q, Q=0...1535	QINT_TQCTL[Q]	Transmit Queue Interrupt Cause Control	Section 10.2.1.10.15
0x00088080	PFINT_GPIO_ENA	PF General Purpose IO Interrupt Enablement	Section 10.2.1.10.16
0x00088188	EMPINT_GPIO_ENA	EMP General Purpose IO Interrupt Enablement	Section 10.2.1.10.17



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x0002BC00 + 0x4*VF, VF=0...127	VFINT_ICR0[VF]	VF Interrupt Zero Cause	Section 10.2.1.10.18
0x0002C000 + 0x4*VF, VF=0...127	VFINT_ICR0_ENA[VF]	VF Interrupt Zero Cause Enablement	Section 10.2.1.10.19
0x0002A400 + 0x4*VF, VF=0...127	VFINT_DYN_CTL0[VF]	VF Interrupt Zero Dynamic Control	Section 10.2.1.10.20
0x0002A000 + 0x4*VF, VF=0...127	VFINT_STAT_CTL0[VF]	VF Interrupt Zero Static Control	Section 10.2.1.10.21
0x0002A800 + 0x4*VF, VF=0...127	VPINT_LNKLST0[VF]	Protected VF Interrupt Zero Linked List	Section 10.2.1.10.22
0x00024800 + 0x4*INTVF, INTVF=0...511	VFINT_DYN_CTLN[INTVF]	VF Interrupt N Dynamic Control	Section 10.2.1.10.23
0x00025000 + 0x4*INTVF, INTVF=0...511	VPINT_LNKLSTN[INTVF]	Protected VF Interrupt N Linked List	Section 10.2.1.10.24
0x00028000 + 0x400*n + 0x4*VF, n=0...2, VF=0...127	VFINT_ITR0[n,VF]	VF Interrupt Throttling for Interrupt Zero	Section 10.2.1.10.25
0x00020000 + 0x800*n + 0x4*INTVF, n=0...2, INTVF=0...511	VFINT_ITRN[n,INTVF]	VF Interrupt Throttling for Interrupt N	Section 10.2.1.10.26
0x0002AC00 + 0x4*VF, VF=0...127	VPINT_RATE0[VF]	Protected VF Interrupt Zero Rate Limit	Section 10.2.1.10.27
0x00025800 + 0x4*INTVF, INTVF=0...511	VPINT_RATEN[INTVF]	Protected VF Interrupt N Rate Limit	Section 10.2.1.10.28
PF - Virtualization PF Registers			
0x000E6000 + 0x4*VF, VF=0...127	VP_MDET_TX[VF]	Malicious Driver Detected on Tx	Section 10.2.1.11.5
0x000E6400	PF_MDET_TX	Malicious Driver Detected on Tx	Section 10.2.1.11.7
0x000E6480	GL_MDET_TX	Malicious Driver Tx Event Details	Section 10.2.1.11.6
0x0012A400	PF_MDET_RX	Malicious Driver Detected on Rx	Section 10.2.1.11.3
0x0012A000 + 0x4*VF, VF=0...127	VP_MDET_RX[VF]	Malicious Driver Detected on Rx	Section 10.2.1.11.1
0x0012A510	GL_MDET_RX	Malicious Driver Rx Event Details	Section 10.2.1.11.2
0x001C0500	PF_VT_PFALLOC	PF Resources Allocation	Section 10.2.1.11.10
PF - DCB Registers			
0x00083000	PRTDCB_GENC	Port DCB General Control	Section 10.2.1.12.3
0x00083044	GLDCB_GENC	Global DCB General Control	Section 10.2.1.12.1
0x00083020	PRTDCB_GENS	Port DCB General Status	Section 10.2.1.12.2
0x001C0980	PRTDCB_TC2PFC	DCB TC to PFC Mapping	Section 10.2.1.12.4
0x001C09A0	PRTDCB_RUP2TC	DCB Receive UP to TC Mapping for RCB	Section 10.2.1.12.5
0x000A21A0	PRTDCB_TCPMC	DCB Transmit Command Pipe Monitor Control	Section 10.2.1.12.6
0x000A2040 + 0x20*n, n=0...7	PRTDCB_TCWSTC[n]	DCB Transmit Command Waiting Status per TC	Section 10.2.1.12.7
0x000A0180	PRTDCB_TDPMC	DCB Transmit Data Pipe Monitor Control	Section 10.2.1.12.8
0x000AE060	PRTDCB_TETSC_TCB	DCB Transmit ETS Control for TCB	Section 10.2.1.12.9



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00098060	PRTDCB_TETSC_TPB	DCB Transmit ETS Control for TPB	Section 10.2.1.12.10
0x000A0040 + 0x20*n, n=0...7	PRTDCB_TCMSTC[n]	DCB Transmit Frame Monitoring Status per TC	Section 10.2.1.12.11
0x001E4660 + 0x20*n, n=0...7	PRTDCB_TPFCTS[n]	DCB Transmit PFC Timer Status	Section 10.2.1.12.12
0x001E4560	PRTDCB_TFCS	Transmit Flow Control Status	Section 10.2.1.12.13
0x001E2400	PRTDCB_MFLCN	MAC Flow Control Register	Section 10.2.1.12.14
0x001E4640	PRTDCB_FCCFG	Flow Control Configuration	Section 10.2.1.12.15
0x001E4600	PRTDCB_FCRTV	Flow Control Refresh Threshold Value	Section 10.2.1.12.16
0x001E4580 + 0x20*n, n=0...3	PRTDCB_FCTTVN[n]	Flow Control Transmit Timer Value n	Section 10.2.1.12.17
0x001223E0	PRTDCB_RETSC	DCB Receive ETS Control	Section 10.2.1.12.18
0x00122180 + 0x20*n, n=0...7	PRTDCB_RETSTCC[n]	DCB Receive ETS per TC Control	Section 10.2.1.12.19
0x001223A0	PRTDCB_RPPMC	DCB Receive per Port Pipe Monitor Control	Section 10.2.1.12.20
0x001C0B00	PRTDCB_RUP	DCB Receive UP in PPRS	Section 10.2.1.12.21
0x00122618	GLDCB_RUPTI	DCB Receive per UP PFC Timer Indication	Section 10.2.1.12.22
0x00122400 + 0x20*n, n=0...7	PRTDCB_RUPTQ[n]	DCB Receive per UP PFC Timer Queue	Section 10.2.1.12.23
PF - Receive Packet Buffer Registers			
0x000AC830	GLRPB_GHW	RPB Global High Watermark	Section 10.2.1.13.1
0x000AC834	GLRPB_GLW	RPB Global Low Watermark	Section 10.2.1.13.2
0x000AC844	GLRPB_PHW	RPB Packet High Watermark	Section 10.2.1.13.3
0x000AC848	GLRPB_PLW	RPB Packet Low Watermark	Section 10.2.1.13.4
0x000AC828	GLRPB_DPSS	RPB Dedicated Pool Size for Single Shared Buffer State	Section 10.2.1.13.5
0x000AC100 + 0x20*n, n=0...7	PRTRPB_DHW[n]	RPB Dedicated Pool High Watermark	Section 10.2.1.13.6
0x000AC220 + 0x20*n, n=0...7	PRTRPB_DLW[n]	RPB Dedicated Pool Low Watermark	Section 10.2.1.13.7
0x000AC320 + 0x20*n, n=0...7	PRTRPB_DPS[n]	RPB Dedicated Pool Size	Section 10.2.1.13.8
0x000AC480 + 0x20*n, n=0...7	PRTRPB_SHT[n]	RPB Shared Pool High Threshold	Section 10.2.1.13.9
0x000AC5A0 + 0x20*n, n=0...7	PRTRPB_SLT[n]	RPB Shared Pool Low Threshold	Section 10.2.1.13.10
0x000AC7C0	PRTRPB_SPS	RPB Shared Pool Size	Section 10.2.1.13.11
0x000AC6A0	PRTRPB_SLW	RPB Shared Pool Low Watermark	Section 10.2.1.13.13
0x000AC580	PRTRPB_SHW	RPB Shared Pool High Watermark	Section 10.2.1.13.12
PF - HMC Registers			
0x000C2004	GLHMC_LANTXOBSZ	Private Memory LAN Tx Object Size	Section 10.2.1.14.1
0x000C2008	GLHMC_LANQMAX	Private Memory LAN Queue Maximum	Section 10.2.1.14.2



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x000C200C	GLHMC_LANRXOBJSZ	Private Memory LAN Rx Object Size	Section 10.2.1.14.3
0x000C0800 + 0x4*n, n=0...15	GLHMC_SDPART[n]	Private Memory Segment Table Partitioning Registers	Section 10.2.1.14.4
0x000C0000	PFHMC_SDCMD	Private Memory Space Segment Descriptor Command	Section 10.2.1.14.5
0x000C0100	PFHMC_SDDATALOW	Private Memory Space Segment Descriptor Data Low	Section 10.2.1.14.6
0x000C0200	PFHMC_SDDATAHIGH	Private Memory Space Segment Descriptor Data High	Section 10.2.1.14.7
0x000C0300	PFHMC_PDINV	Private Memory Space Page Descriptor Invalidate	Section 10.2.1.14.8
0x000C0400	PFHMC_ERRORINFO	Host Memory Cache Error Information Register	Section 10.2.1.14.9
0x000C0500	PFHMC_ERRORDATA	Host Memory Cache Error Data Register	Section 10.2.1.14.10
0x000C6200 + 0x4*n, n=0...15	GLHMC_LANTXBASE[n]	FPM LAN TX Queue Base	Section 10.2.1.14.11
0x000C6300 + 0x4*n, n=0...15	GLHMC_LANTXCNT[n]	FPM LAN TX Queue Object Count	Section 10.2.1.14.12
0x000C6400 + 0x4*n, n=0...15	GLHMC_LANRXBASE[n]	FPM LAN RX Queue Base	Section 10.2.1.14.13
0x000C6500 + 0x4*n, n=0...15	GLHMC_LANRXCNT[n]	FPM LAN RX Queue Object Count	Section 10.2.1.14.14
PF - CM Registers			
0x0010C080	PFCM_LAN_ERRDATA	CMLAN Error Data	Section 10.2.1.15.1
0x0010C000	PFCM_LAN_ERRINFO	CMLAN Error Information	Section 10.2.1.15.2
0x0010C300	PFCM_LANCTXCTL	CMLAN Context Control Register	Section 10.2.1.15.3
0x0010C380	PFCM_LANCTXSTAT	CMLAN Context Status Register	Section 10.2.1.15.4
0x0010C100 + 0x80*n, n=0...3	PFCM_LANCTXDATA[n]	CMLAN Context Data Registers	Section 10.2.1.15.5
PF - Admin Queue			
0x00080000	PF_ATQBAL	PF Admin Transmit Queue Base Address Low	Section 10.2.1.16.1
0x00080100	PF_ATQBAH	PF Admin Transmit Queue Base Address High	Section 10.2.1.16.2
0x00080200	PF_ATQLEN	PF Admin Transmit Queue Length	Section 10.2.1.16.3
0x00080300	PF_ATQH	PF Admin Transmit Head	Section 10.2.1.16.4
0x00080400	PF_ATQT	PF Admin Transmit Tail	Section 10.2.1.16.5
0x00080080	PF_ARQBAL	PF Admin Receive Queue Base Address Low	Section 10.2.1.16.6
0x00080180	PF_ARQBAH	PF Admin Receive Queue Base Address High	Section 10.2.1.16.7
0x00080280	PF_ARQLEN	PF Admin Receive Queue Length	Section 10.2.1.16.8
0x00080380	PF_ARQH	PF Admin Receive Queue Head	Section 10.2.1.16.9
0x00080480	PF_ARQT	PF Admin Receive Queue Tail	Section 10.2.1.16.10
0x00080800 + 0x4*VF, VF=0...127	VF_ATQBAL[VF]	VF Admin Transmit Queue Base Address Low	Section 10.2.1.16.11



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00081000 + 0x4*VF, VF=0...127	VF_ATQBAH[VF]	VF Admin Transmit Queue Base Address High	Section 10.2.1.16.12
0x00081800 + 0x4*VF, VF=0...127	VF_ATQLEN[VF]	VF Admin Transmit Queue Length	Section 10.2.1.16.13
0x00082000 + 0x4*VF, VF=0...127	VF_ATQH[VF]	VF Admin Transmit Head	Section 10.2.1.16.14
0x00082800 + 0x4*VF, VF=0...127	VF_ATQT[VF]	VF Admin Transmit Tail	Section 10.2.1.16.15
0x00080C00 + 0x4*VF, VF=0...127	VF_ARQBAL[VF]	VF Admin Receive Queue Base Address Low	Section 10.2.1.16.16
0x00081400 + 0x4*VF, VF=0...127	VF_ARQBAH[VF]	VF Admin Receive Queue Base Address High	Section 10.2.1.16.17
0x00081C00 + 0x4*VF, VF=0...127	VF_ARQLEN[VF]	VF Admin Receive Queue Length	Section 10.2.1.16.18
0x00082400 + 0x4*VF, VF=0...127	VF_ARQH[VF]	VF Admin Receive Queue Head	Section 10.2.1.16.19
0x00082C00 + 0x4*VF, VF=0...127	VF_ARQT[VF]	VF Admin Receive Queue Tail	Section 10.2.1.16.20
0x00080040	GL_ATQBAL	Global Admin Transmit Queue Base Address Low	Section 10.2.1.16.21
0x00080140	GL_ATQBAH	Global Admin Transmit Queue Base Address High	Section 10.2.1.16.22
0x00080240	GL_ATQLEN	Global Admin Transmit Queue Length	Section 10.2.1.16.23
0x00080340	GL_ATQH	Global Admin Transmit Head	Section 10.2.1.16.24
0x00080440	GL_ATQT	Global Admin Transmit Tail	Section 10.2.1.16.25
0x000800C0	GL_ARQBAL	Global Admin Receive Queue Base Address Low	Section 10.2.1.16.26
0x000801C0	GL_ARQBAH	Global Admin Receive Queue Base Address High	Section 10.2.1.16.27
0x000803C0	GL_ARQH	Global Admin Receive Queue Head	Section 10.2.1.16.28
0x000804C0	GL_ARQT	Global Admin Receive Queue Tail	Section 10.2.1.16.29
PF - Statistics Registers			
0x00300000 + 0x8*n, n=0...3	GLPRT_GORCL[n]	Port Good Octets Received Count Low	Section 10.2.1.17.1
0x00300004 + 0x8*n, n=0...3	GLPRT_GORCH[n]	Port Good Octets Received Count High	Section 10.2.1.17.2
0x003005A0 + 0x8*n, n=0...3	GLPRT_UPRCL[n]	Port Unicast Packets Received Count Low	Section 10.2.1.17.3
0x003005A4 + 0x8*n, n=0...3	GLPRT_UPRCH[n]	Port Unicast Packets Received Count High	Section 10.2.1.17.4
0x003005C0 + 0x8*n, n=0...3	GLPRT_MPRCL[n]	Port Multicast Packets Received Count Low	Section 10.2.1.17.5
0x003005C4 + 0x8*n, n=0...3	GLPRT_MPRCH[n]	Port Multicast Packets Received Count High	Section 10.2.1.17.6
0x003005E0 + 0x8*n, n=0...3	GLPRT_BPRCL[n]	Port Broadcast Packets Received Count Low	Section 10.2.1.17.7
0x003005E4 + 0x8*n, n=0...3	GLPRT_BPRCH[n]	Port Broadcast Packets Received Count High	Section 10.2.1.17.8
0x00300600 + 0x8*n, n=0...3	GLPRT_RDPC[n]	Port Receive Packets Discarded Count	Section 10.2.1.17.9
0x00300660 + 0x8*n, n=0...3	GLPRT_RUPP[n]	Port Received With No Destination	Section 10.2.1.17.10



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00300680 + 0x8*n, n=0...3	GLPRT_GOTCL[n]	Port Good Octets Transmit Count Low	Section 10.2.1.17.11
0x00300684 + 0x8*n, n=0...3	GLPRT_GOTCH[n]	Port Good Octets transmit Count High	Section 10.2.1.17.12
0x003009C0 + 0x8*n, n=0...3	GLPRT_UPTCL[n]	Port Unicast Packets Transmit Count Low	Section 10.2.1.17.13
0x003009C4 + 0x8*n, n=0...3	GLPRT_UPTCH[n]	Port Unicast Packets Transmit Count High	Section 10.2.1.17.14
0x003009E0 + 0x8*n, n=0...3	GLPRT_MPTCL[n]	Port Multicast Packets Transmit Count Low	Section 10.2.1.17.15
0x003009E4 + 0x8*n, n=0...3	GLPRT_MPTCH[n]	Port Multicast Packets Transmit Count High	Section 10.2.1.17.16
0x00300A00 + 0x8*n, n=0...3	GLPRT_BPTCL[n]	Port Broadcast Packets Transmit Count Low	Section 10.2.1.17.17
0x00300A04 + 0x8*n, n=0...3	GLPRT_BPTCH[n]	Port Broadcast Packets Transmit Count High	Section 10.2.1.17.18
0x00300A20 + 0x8*n, n=0...3	GLPRT_TDOLD[n]	Transmit Discard On Link Down	Section 10.2.1.17.19
0x00300480 + 0x8*n, n=0...3	GLPRT_PRC64L[n]	Packets Received [64 Bytes] Count Low	Section 10.2.1.17.20
0x00300484 + 0x8*n, n=0...3	GLPRT_PRC64H[n]	Packets Received [64 Bytes] Count High	Section 10.2.1.17.21
0x003004A0 + 0x8*n, n=0...3	GLPRT_PRC127L[n]	Packets Received [65-127 Bytes] Count Low	Section 10.2.1.17.22
0x003004A4 + 0x8*n, n=0...3	GLPRT_PRC127H[n]	Packets Received [65-127 Bytes] Count High	Section 10.2.1.17.23
0x003004C0 + 0x8*n, n=0...3	GLPRT_PRC255L[n]	Packets Received [128-255 Bytes] Count Low	Section 10.2.1.17.24
0x003004C4 + 0x8*n, n=0...3	GLPRT_PRC255H[n]	Packets Received [128-255 Bytes] Count High	Section 10.2.1.17.25
0x003004E0 + 0x8*n, n=0...3	GLPRT_PRC511L[n]	Packets Received [256-511 Bytes] Count Low	Section 10.2.1.17.26
0x003004E4 + 0x8*n, n=0...3	GLPRT_PRC511H[n]	Packets Received [256-511 Bytes] Count High	Section 10.2.1.17.27
0x00300500 + 0x8*n, n=0...3	GLPRT_PRC1023L[n]	Packets Received [512-1023 Bytes] Count Low	Section 10.2.1.17.28
0x00300504 + 0x8*n, n=0...3	GLPRT_PRC1023H[n]	Packets Received [512-1023 Bytes] Count High	Section 10.2.1.17.29
0x00300520 + 0x8*n, n=0...3	GLPRT_PRC1522L[n]	Packets Received [1024-1522] Count Low	Section 10.2.1.17.30
0x00300524 + 0x8*n, n=0...3	GLPRT_PRC1522H[n]	Packets Received [1024-1522] Count High	Section 10.2.1.17.31
0x00300540 + 0x8*n, n=0...3	GLPRT_PRC9522L[n]	Packets Received [1523-9522 Bytes] Count Low	Section 10.2.1.17.32
0x00300544 + 0x8*n, n=0...3	GLPRT_PRC9522H[n]	Packets Received [1523-9522 Bytes] Count High	Section 10.2.1.17.33
0x003006A0 + 0x8*n, n=0...3	GLPRT_PTC64L[n]	Packets Transmitted (64 Bytes) Count Low	Section 10.2.1.17.34
0x003006A4 + 0x8*n, n=0...3	GLPRT_PTC64H[n]	Packets Transmitted (64 Bytes) Count High	Section 10.2.1.17.35
0x003006C0 + 0x8*n, n=0...3	GLPRT_PTC127L[n]	Packets Transmitted [65-127 Bytes] Count Low	Section 10.2.1.17.36



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x003006C4 + 0x8*n, n=0...3	GLPRT_ PTC127H[n]	Packets Transmitted [65-127 Bytes] Count High	Section 10.2.1.17.37
0x003006E0 + 0x8*n, n=0...3	GLPRT_PTC255L[n]	Packets Transmitted [128-255 Bytes] Count Low	Section 10.2.1.17.38
0x003006E4 + 0x8*n, n=0...3	GLPRT_ PTC255H[n]	Packets Transmitted [128-255 Bytes] Count High	Section 10.2.1.17.39
0x00300700 + 0x8*n, n=0...3	GLPRT_PTC511L[n]	Packets Transmitted [256-511 Bytes] Count Low	Section 10.2.1.17.40
0x00300704 + 0x8*n, n=0...3	GLPRT_ PTC511H[n]	Packets Transmitted [256-511 Bytes] Count High	Section 10.2.1.17.41
0x00300720 + 0x8*n, n=0...3	GLPRT_ PTC1023L[n]	Packets Transmitted [512-1023 Bytes] Count Low	Section 10.2.1.17.42
0x00300724 + 0x8*n, n=0...3	GLPRT_ PTC1023H[n]	Packets Transmitted [512-1023 Bytes] Count High	Section 10.2.1.17.43
0x00300740 + 0x8*n, n=0...3	GLPRT_ PTC1522L[n]	Packets Transmitted [1024-1522 Bytes] Count Low	Section 10.2.1.17.44
0x00300744 + 0x8*n, n=0...3	GLPRT_ PTC1522H[n]	Packets Transmitted [1024-1522 Bytes] Count High	Section 10.2.1.17.45
0x00300760 + 0x8*n, n=0...3	GLPRT_ PTC9522L[n]	Packets Transmitted [1523-9522 bytes] Count Low	Section 10.2.1.17.46
0x00300764 + 0x8*n, n=0...3	GLPRT_ PTC9522H[n]	Packets Transmitted [1523-9522 bytes] Count High	Section 10.2.1.17.47
0x00300140 + 0x8*n, n=0...3	GLPRT_ LXONRXC[n]	Port Link XON Received Count	Section 10.2.1.17.48
0x00300160 + 0x8*n, n=0...3	GLPRT_ LXOFFRXC[n]	Port Link XOFF Received Count	Section 10.2.1.17.49
0x00300980 + 0x8*n, n=0...3	GLPRT_ LXONTXC[n]	Port Link XON Transmitted Count	Section 10.2.1.17.50
0x003009A0 + 0x8*n, n=0...3	GLPRT_ LXOFFTXC[n]	Port Link XOFF Transmitted Count	Section 10.2.1.17.51
0x00300180 + 0x8*n + 0x20*m, n=0...3, m=0...7	GLPRT_ PXONRXC[n,m]	Priority XON Received Count	Section 10.2.1.17.52
0x00300880 + 0x8*n + 0x20*m, n=0...3, m=0...7	GLPRT_ PXOFFTXC[n,m]	Priority XOFF Transmitted Count	Section 10.2.1.17.53
0x00300780 + 0x8*n + 0x20*m, n=0...3, m=0...7	GLPRT_ PXONTXC[n,m]	Priority XON Transmitted Count	Section 10.2.1.17.54
0x00300280 + 0x8*n + 0x20*m, n=0...3, m=0...7	GLPRT_ PXOFFRXC[n,m]	Priority XOFF Received Count	Section 10.2.1.17.55
0x00300380 + 0x8*n + 0x20*m, n=0...3, m=0...7	GLPRT_ RXON2OFFCNT[n,m]	Priority XON to XOFF Count	Section 10.2.1.17.56
0x00300080 + 0x8*n, n=0...3	GLPRT_ CRCERRS[n]	Port CRC Error Count	Section 10.2.1.17.57
0x003000E0 + 0x8*n, n=0...3	GLPRT_ILLERRC[n]	Port Illegal Byte Error Count	Section 10.2.1.17.58
0x003000C0 + 0x8*n, n=0...3	GLPRT_ERRBC[n]	Port Error Byte Count	Section 10.2.1.17.59
0x00300020 + 0x8*n, n=0...3	GLPRT_MLFC[n]	Port MAC Local Fault Count	Section 10.2.1.17.60
0x00300040 + 0x8*n, n=0...3	GLPRT_MRFC[n]	Port MAC Remote Fault Count	Section 10.2.1.17.61
0x003000A0 + 0x8*n, n=0...3	GLPRT_RLECC[n]	Receive Length Error Count	Section 10.2.1.17.62



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00300100 + 0x8*n, n=0...3	GLPRT_RUC[n]	Receive Undersize Count	Section 10.2.1.17.63
0x00300560 + 0x8*n, n=0...3	GLPRT_RFC[n]	Receive Fragment Count	Section 10.2.1.17.64
0x00300120 + 0x8*n, n=0...3	GLPRT_ROC[n]	Receive Oversize Count	Section 10.2.1.17.65
0x00300580 + 0x8*n, n=0...3	GLPRT_RJC[n]	Receive Jabber Count	Section 10.2.1.17.66
0x00300060 + 0x8*n, n=0...3	GLPRT_MSPDC[n]	Port MAC Short Packet Discard Count	Section 10.2.1.17.67
0x00300620 + 0x8*n, n=0...3	GLPRT_LDPC[n]	Loopback Packets Discarded Count	Section 10.2.1.17.68
0x0035C000 + 0x8*n, n=0...15	GLSW_GORCL[n]	Switch Good Octets Received Count Low	Section 10.2.1.17.69
0x0035C004 + 0x8*n, n=0...15	GLSW_GORCH[n]	Switch Good Octets Received Count High	Section 10.2.1.17.70
0x00370000 + 0x8*n, n=0...15	GLSW_UPRCL[n]	Switch Unicast Packets Received Count Low	Section 10.2.1.17.71
0x00370004 + 0x8*n, n=0...15	GLSW_UPRCH[n]	Switch Unicast Packets Received Count High	Section 10.2.1.17.72
0x00370080 + 0x8*n, n=0...15	GLSW_MPRCL[n]	Switch Multicast Packets Received Count Low	Section 10.2.1.17.73
0x00370084 + 0x8*n, n=0...15	GLSW_MPRCH[n]	Switch Multicast Packets Received Count High	Section 10.2.1.17.74
0x00370100 + 0x8*n, n=0...15	GLSW_BPRCL[n]	Switch Broadcast Packets Received Count Low	Section 10.2.1.17.75
0x00370104 + 0x8*n, n=0...15	GLSW_BPRCH[n]	Switch Broadcast Packets Received Count High	Section 10.2.1.17.76
0x00348000 + 0x8*n, n=0...15	GLSW_TDPC[n]	Switch Transmit Packets Discarded Count	Section 10.2.1.17.77
0x00370180 + 0x8*n, n=0...15	GLSW_RUPP[n]	Switch Received Unknown Packet Protocol Count	Section 10.2.1.17.78
0x0032C000 + 0x8*n, n=0...15	GLSW_GOTCL[n]	Switch Good Octets transmit Count Low	Section 10.2.1.17.79
0x0032C004 + 0x8*n, n=0...15	GLSW_GOTCH[n]	Switch Good Octets transmit Count High	Section 10.2.1.17.80
0x00340000 + 0x8*n, n=0...15	GLSW_UPTCL[n]	Switch Unicast Packets Transmit Count Low	Section 10.2.1.17.81
0x00340004 + 0x8*n, n=0...15	GLSW_UPTCH[n]	Switch Unicast Packets Transmit Count High	Section 10.2.1.17.82
0x00340080 + 0x8*n, n=0...15	GLSW_MPTCL[n]	Switch Multicast Packets Transmit Count Low	Section 10.2.1.17.83
0x00340084 + 0x8*n, n=0...15	GLSW_MPTCH[n]	Switch Multicast Packets Transmit Count High	Section 10.2.1.17.84
0x00340100 + 0x8*n, n=0...15	GLSW_BPTCL[n]	Switch Broadcast Packets Transmit Count Low	Section 10.2.1.17.85
0x00340104 + 0x8*n, n=0...15	GLSW_BPTCH[n]	Switch Broadcast Packets Transmit Count High	Section 10.2.1.17.86
0x00360000 + 0x8*n, n=0...127	GLVEBVL_GORCL[n]	VEB VLAN Receive Byte Count Low	Section 10.2.1.17.87
0x00360004 + 0x8*n, n=0...127	GLVEBVL_GORCH[n]	VEB VLAN Receive Byte Count High	Section 10.2.1.17.88



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00330000 + 0x8*n, n=0...127	GLVEBVL_ GOTCL[n]	VEB VLAN Transmit Byte Count Low	Section 10.2.1.17.89
0x00330004 + 0x8*n, n=0...127	GLVEBVL_ GOTCH[n]	VEB VLAN Transmit Byte Count High	Section 10.2.1.17.90
0x00374000 + 0x8*n, n=0...127	GLVEBVL_UPCL[n]	VEB VLAN Unicast Packet Count Low	Section 10.2.1.17.91
0x00374004 + 0x8*n, n=0...127	GLVEBVL_UPCH[n]	VEB VLAN Unicast Packet Count High	Section 10.2.1.17.92
0x00374400 + 0x8*n, n=0...127	GLVEBVL_MPCL[n]	VEB VLAN Multicast Packet Count Low	Section 10.2.1.17.93
0x00374404 + 0x8*n, n=0...127	GLVEBVL_MPCH[n]	VEB VLAN Multicast Packet Count High	Section 10.2.1.17.94
0x00374800 + 0x8*n, n=0...127	GLVEBVL_BPCL[n]	VEB VLAN Broadcast Packet Count Low	Section 10.2.1.17.95
0x00374804 + 0x8*n, n=0...127	GLVEBVL_BPCH[n]	VEB VLAN Broadcast Packet Count High	Section 10.2.1.17.96
0x00368000 + 0x8*n + 0x40*m, n=0...7, m=0...15	GLVEBTC_ RPCL[n,m]	VEB TC Receive Packet Count Low	Section 10.2.1.17.97
0x00368004 + 0x8*n + 0x40*m, n=0...7, m=0...15	GLVEBTC_ RPCH[n,m]	VEB TC Receive Packet Count High	Section 10.2.1.17.98
0x00364000 + 0x8*n + 0x40*m, n=0...7, m=0...15	GLVEBTC_ RBCL[n,m]	VEB TC Receive Byte Count Low	Section 10.2.1.17.99
0x00364004 + 0x8*n + 0x40*m, n=0...7, m=0...15	GLVEBTC_ RBCH[n,m]	VEB TC Receive Byte Count High	Section 10.2.1.17.100
0x00338000 + 0x8*n + 0x40*m, n=0...7, m=0...15	GLVEBTC_ TPCL[n,m]	VEB TC Transmit Packet Count Low	Section 10.2.1.17.101
0x00338004 + 0x8*n + 0x40*m, n=0...7, m=0...15	GLVEBTC_ TPCH[n,m]	VEB TC Transmit Packet Count High	Section 10.2.1.17.102
0x00334000 + 0x8*n + 0x40*m, n=0...7, m=0...15	GLVEBTC_ TBCL[n,m]	VEB TC Transmit Byte Count Low	Section 10.2.1.17.103
0x00334004 + 0x8*n + 0x40*m, n=0...7, m=0...15	GLVEBTC_ TBCH[n,m]	VEB TC Transmit Byte Count High	Section 10.2.1.17.104
0x00358000 + 0x8*n, n=0...383	GLV_GORCL[n]	VSI Good Octets Received Count Low	Section 10.2.1.17.105
0x00358004 + 0x8*n, n=0...383	GLV_GORCH[n]	VSI Good Octets Received Count High	Section 10.2.1.17.106
0x0036C000 + 0x8*n, n=0...383	GLV_UPRCL[n]	VSI Unicast Packets Received Count Low	Section 10.2.1.17.107
0x0036C004 + 0x8*n, n=0...383	GLV_UPRCH[n]	VSI Unicast Packets Received Count High	Section 10.2.1.17.108
0x0036CC00 + 0x8*n, n=0...383	GLV_MPRCL[n]	VSI Multicast Packets Received Count Low	Section 10.2.1.17.109
0x0036CC04 + 0x8*n, n=0...383	GLV_MPRCH[n]	VSI Multicast Packets Received Count High	Section 10.2.1.17.110
0x0036D800 + 0x8*n, n=0...383	GLV_BPRCL[n]	VSI Broadcast Packets Received Count Low	Section 10.2.1.17.111
0x0036D804 + 0x8*n, n=0...383	GLV_BPRCH[n]	VSI Broadcast Packets Received Count High	Section 10.2.1.17.112
0x00310000 + 0x8*n, n=0...383	GLV_RDPC[n]	VSI Received Discard Packet Count	Section 10.2.1.17.113
0x0036E400 + 0x8*n, n=0...383	GLV_RUPP[n]	VSI Received Unknown Packet Protocol Count	Section 10.2.1.17.114



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00328000 + 0x8*n, n=0...383	GLV_GOTCL[n]	VSI Good Octets Transmit Count Low	Section 10.2.1.17.115
0x00328004 + 0x8*n, n=0...383	GLV_GOTCH[n]	VSI Good Octets Transmit Count High	Section 10.2.1.17.116
0x0033C000 + 0x8*n, n=0...383	GLV_UPTCL[n]	VSI Unicast Packets Transmit Count Low	Section 10.2.1.17.117
0x0033C004 + 0x8*n, n=0...383	GLV_UPTCH[n]	VSI Unicast Packets Transmit Count High	Section 10.2.1.17.118
0x0033CC00 + 0x8*n, n=0...383	GLV_MPTCL[n]	VSI Multicast Packets Transmit Count Low	Section 10.2.1.17.119
0x0033CC04 + 0x8*n, n=0...383	GLV_MPTCH[n]	VSI Multicast Packets Transmit Count High	Section 10.2.1.17.120
0x0033D800 + 0x8*n, n=0...383	GLV_BPTCL[n]	VSI Broadcast Packets Transmit Count Low	Section 10.2.1.17.121
0x0033D804 + 0x8*n, n=0...383	GLV_BPTCH[n]	VSI Broadcast Packets Transmit Count High	Section 10.2.1.17.122
0x00344000 + 0x8*n, n=0...383	GLV_TEPC[n]	VSI Transmit Error Packet Count	Section 10.2.1.17.123
PF - LAN Transmit Receive Registers			
0x00051060	GL_RDPU_CNTRL	Receive Processing Block Control	Section 10.2.1.18.1
0x001C0400	PFLAN_QALLOC	PF Queue Allocation	Section 10.2.1.18.2
0x00074000 + 0x4*VF, VF=0...127	VPLAN_MAPENA[VF]	VF LAN Enablement	Section 10.2.1.18.3
0x0012A500	GLLAN_RCTL_0	Global RLAN Control 0	Section 10.2.1.18.4
0x000442D8	GLLAN_TSOMSK_F	Global TSO TCP Mask First	Section 10.2.1.18.5
0x000442DC	GLLAN_TSOMSK_M	Global TSO TCP Mask Middle	Section 10.2.1.18.6
0x000442E0	GLLAN_TSOMSK_L	Global TSO TCP Mask Last	Section 10.2.1.18.7
0x00104000 + 0x4*Q, Q=0...1535	QTX_CTL[Q]	Global Transmit Queue Control	Section 10.2.1.18.8
0x000E6500 + 0x4*n, n=0...11	GLLAN_TXPRE_QDIS[n]	Global Transmit Pre Queue Disable	Section 10.2.1.18.9
0x00100000 + 0x4*Q, Q=0...1535	QTX_ENA[Q]	Global Transmit Queue Enable	Section 10.2.1.18.10
0x000E4000 + 0x4*Q, Q=0...1535	QTX_HEAD[Q]	Global Transmit Queue Head	Section 10.2.1.18.11
0x00108000 + 0x4*Q, Q=0...1535	QTX_TAIL[Q]	Global Transmit Queue Tail	Section 10.2.1.18.12
0x00120000 + 0x4*Q, Q=0...1535	QRX_ENA[Q]	Global Receive Queue Enable	Section 10.2.1.18.13
0x00128000 + 0x4*Q, Q=0...1535	QRX_TAIL[Q]	Global Receive Queue Tail	Section 10.2.1.18.14
0x00070000 + 0x400*n + 0x4*VF, n=0...15, VF=0...127	VPLAN_QTABLE[n,VF]	VF PF Queue Mapping Table	Section 10.2.1.18.15
0x0020C800 + 0x4*VSI, VSI=0...383	VSILAN_QBASE[VSI]	VSI Queue Control	Section 10.2.1.18.16
0x00200000 + 0x800*n + 0x4*VSI, n=0...7, VSI=0...383	VSILAN_QTABLE[n,VSI]	VSI Receive Queue Mapping Table	Section 10.2.1.18.17



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
PF - Rx Filters Registers			
0x00256E60	PRTQF_CTL_0	Port Queue Filter Control 0	Section 10.2.1.19.1
0x00269BA4	GLQF_CTL	Global Queue Filter Control	Section 10.2.1.19.2
0x001C0AC0	PFQF_CTL_0	PF Queue Filter Control 0	Section 10.2.1.19.3
0x00245D80	PFQF_CTL_1	PF Queue Filter Control 1	Section 10.2.1.19.3
0x00267E00 + 0x4*n + 0x8*m, n=0...1, m=0...63	GLQF_SWAP[n,m]	Global Queue Filter SWAP Fields	Section 10.2.1.19.4
0x00269D00 + 0x4*n, n=0...63	GLQF_HSYM[n]	Global Queue Filter Symmetric Hash Enablement	Section 10.2.1.19.5
0x00268900 + 0x4*n, n=0...63	GLQF_ORT[n]	Global Queue Filter - Offset Redirection Table	Section 10.2.1.19.6
0x00268C80 + 0x4*n, n=0...23	GLQF_PIT[n]	Global Queue Filter - Parser Information Table	Section 10.2.1.19.6
0x00255200 + 0x20*n, n=0...8	PRTQF_FLX_PIT[n]	Port Queue Filter - Flexible Parser Information Table	Section 10.2.1.19.6
0x00269760 + 0x4*n, n=0...3	GL_PRS_FVBM[n]	Global Tunneling Key Mask	Section 10.2.1.19.7
0x00250000 + 0x40*n + 0x20*m, n=0...63, m=0...1	PRTQF_FD_INSET[n,m]	Port Queue Filter Flow Director Input Set	Section 10.2.1.19.7
0x00253800 + 0x20*n, n=0...63	PRTQF_FD_FLXINSET[n]	Port Queue Filter Flow Director Input Set	Section 10.2.1.19.7
0x00267600 + 0x4*n + 0x8*m, n=0...1, m=0...63	GLQF_HASH_INSET[n,m]	Global Queue Filter Hash Input Set	Section 10.2.1.19.8
0x00267200 + 0x4*n + 0x8*m, n=0...1, m=0...63	GLQF_FD_MSK[n,m]	Global Queue Filter Flow Director Mask	Section 10.2.1.19.8
0x00252000 + 0x40*n + 0x20*m, n=0...63, m=0...1	PRTQF_FD_MSK[n,m]	Port Queue Filter Flow Director Mask	Section 10.2.1.19.8
0x00267A00 + 0x4*n + 0x8*m, n=0...1, m=0...63	GLQF_HASH_MSK[n,m]	Global Queue Filter Hash Mask	Section 10.2.1.19.9
0x00245900 + 0x80*n, n=0...1	PFQF_HENA[n]	PF Queue Filter Hash Enabled Packet Type	Section 10.2.1.19.9
0x00230800 + 0x400*n + 0x4*VF, n=0...1, VF=0...127	VFQF_HENA[n,VF]	VF Queue Filter Hash Enabled Packet Type	Section 10.2.1.19.10
0x00245400 + 0x80*n, n=0...7	PFQF_HREGION[n]	PF Queue Filter Hash Region of Queues	Section 10.2.1.19.11
0x0022E000 + 0x400*n + 0x4*VF, n=0...7, VF=0...127	VFQF_HREGION[n,VF]	VF Queue Filter Hash Region of Queues	Section 10.2.1.19.12
0x00206000 + 0x800*n + 0x4*VSI, n=0...3, VSI=0...383	VSIQF_TCREGION[n,VSI]	VSI Receive Traffic Class Queues	Section 10.2.1.19.13
0x00244800 + 0x80*n, n=0...12	PFQF_HKEY[n]	PF Queue Filter Hash Key	Section 10.2.1.19.14
0x00228000 + 0x400*n + 0x4*VF, n=0...12, VF=0...127	VFQF_HKEY[n,VF]	VF Queue Filter Hash Key	Section 10.2.1.19.15
0x00270140 + 0x4*n, n=0...12	GLQF_HKEY[n]	Global Queue Filter Hash Key	Section 10.2.1.19.16
0x00240000 + 0x80*n, n=0...127	PFQF_HLUT[n]	PF Queue Filter Hash LUT	Section 10.2.1.19.17



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00220000 + 0x400*n + 0x4*VF, n=0...15, VF=0...127	VFQF_HLUT[n,VF]	VF Queue Filter Hash LUT	Section 10.2.1.19.18
0x00266800 + 0x4*n, n=0...511	GLQF_PCNT[n]	Global Queue Filter Packet Counter	Section 10.2.1.19.19
0x00269BAC	GLQF_FDCNT_0	Global Queue Filter Flow Director Status 0	Section 10.2.1.19.20
0x00246280	PFQF_FDALLOC	PF Queue Filter Flow Director Allocation	Section 10.2.1.19.21
0x00246380	PFQF_FDSTAT	PF Queue Filter Flow Director Allocation Status	Section 10.2.1.19.22
PF - TimeSync (IEEE 1588) Registers			
0x001E4200	PRTTSYN_CTL0	Port Time Sync Control 0	Section 10.2.1.20.1
0x00085020	PRTTSYN_CTL1	Port Time Sync Control 1	Section 10.2.1.20.2
0x001E4220	PRTTSYN_STAT_0	Port Time Sync Status 0	Section 10.2.1.20.3
0x00085140	PRTTSYN_STAT_1	Port Time Sync Status 1	Section 10.2.1.20.4
0x001E4100	PRTTSYN_TIME_L	Port Time Sync Time Low	Section 10.2.1.20.5
0x001E4120	PRTTSYN_TIME_H	Port Time Sync Time High	Section 10.2.1.20.6
0x001E4040	PRTTSYN_INC_L	Port Time Sync Increment Value Low	Section 10.2.1.20.7
0x001E4060	PRTTSYN_INC_H	Port Time Sync Increment Value High	Section 10.2.1.20.8
0x001E4280	PRTTSYN_ADJ	Port Time Sync Adjustment	Section 10.2.1.20.9
0x000850C0 + 0x20*n, n=0...3	PRTTSYN_RXTIME_L[n]	Port Time Sync Receive PTP Packet Time Low	Section 10.2.1.20.10
0x00085040 + 0x20*n, n=0...3	PRTTSYN_RXTIME_H[n]	port Time Sync Receive PTP Packet Time High	Section 10.2.1.20.11
0x001E41C0	PRTTSYN_TXTIME_L	Port Time Sync Transmit Packet Time Low	Section 10.2.1.20.12
0x001E41E0	PRTTSYN_TXTIME_H	Port Time Sync Transmit Packet Time High	Section 10.2.1.20.13
0x001E42A0 + 0x20*n, n=0...1	PRTTSYN_AUX_0[n]	Port Time Sync AUX Control 0	Section 10.2.1.20.14
0x001E42E0 + 0x20*n, n=0...1	PRTTSYN_AUX_1[n]	Port Time Sync AUX Control 1	Section 10.2.1.20.15
0x001E4140 + 0x20*n, n=0...1	PRTTSYN_TGT_L[n]	Port Time Sync Target Time Low	Section 10.2.1.20.16
0x001E4180 + 0x20*n, n=0...1	PRTTSYN_TGT_H[n]	Port Time Sync Target Time High	Section 10.2.1.20.17
0x001E4240 + 0x20*n, n=0...1	PRTTSYN_CLKO[n]	Port Time Sync Clock Out Duration	Section 10.2.1.20.18
0x001E4080 + 0x20*n, n=0...1	PRTTSYN_EVNT_L[n]	Port Time Sync Event Time Low	Section 10.2.1.20.19
0x001E40C0 + 0x20*n, n=0...1	PRTTSYN_EVNT_H[n]	Port Time Sync Event Time High	Section 10.2.1.20.20
PF - Manageability Registers			
0x00083100	GL_FWRESETCNT	Firmware Reset Count	Section 10.2.1.21.1
0x00256A20	PRT_MNG_MANC	Management Control Register	Section 10.2.1.21.2
0x00255F00 + 0x20*n, n=0...7	PRT_MNG_MDEF_EXT[n]	Manageability Decision Filters	Section 10.2.1.21.3
0x00255D00 + 0x20*n, n=0...7	PRT_MNG_MDEF[n]	Manageability Decision Filters1	Section 10.2.1.21.4



Table 10-7. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x00256A60	PRT_MNG_MNGONLY	Management Only Traffic Register	Section 10.2.1.21.5
0x00256580 + 0x20*n, n=0...3	PRT_MNG_MDEFVSI[n]	Management Decision Filters Buffers	Section 10.2.1.21.6
0x00256480 + 0x20*n, n=0...3	PRT_MNG_MMAL[n]	Manageability MAC Address Low	Section 10.2.1.21.7
0x00256380 + 0x20*n, n=0...3	PRT_MNG_MMAH[n]	Manageability MAC Address High	Section 10.2.1.21.8
0x00255900 + 0x20*n, n=0...7	PRT_MNG_MAVTV[n]	Management VLAN TAG Value	Section 10.2.1.21.9
0x00256780 + 0x20*n, n=0...3	PRT_MNG_METF[n]	Management Ethernet Type Filters	Section 10.2.1.21.10
0x00254200 + 0x20*n, n=0...15	PRT_MNG_MIPAF6[n]	Manageability IPv6 Address Filter	Section 10.2.1.21.11
0x00256280 + 0x20*n, n=0...3	PRT_MNG_MIPAF4[n]	Manageability IPv4 Address Filter	Section 10.2.1.21.12
0x00254E00 + 0x20*n, n=0...15	PRT_MNG_MFUTP[n]	Management Flex UDP/TCP Ports	Section 10.2.1.21.13
0x00256AA0	PRT_MNG_MSFM	Manageability Special Filters Modifiers	Section 10.2.1.21.14
0x000852A0 + 0x20*n, n=0...31	PRT_MNG_FTFT_DATA[n]	Flexible TCO Filter Table Registers - Data	Section 10.2.1.21.15
0x00085160 + 0x20*n, n=0...7	PRT_MNG_FTFT_MASK[n]	Flexible TCO Filter Table Registers - Mask	Section 10.2.1.21.16
0x00085260	PRT_MNG_FTFT_LENGTH	Flexible TCO Filter Table Registers - Length	Section 10.2.1.21.17
0x000B6130	GL_MNG_HWARB_CTRL	Hardware Arbitration Control	Section 10.2.1.21.18
0x000B6134	GL_MNG_FWSM	Firmware Semaphore	Section 10.2.1.21.19

10.2.1.1 General Registers

This category contains registers for general device control and status.

10.2.1.1.1 Global Status - GLGEN_STAT (0x000B612C)

Reset Source: POR

NVM Load: Forbidden Load

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DCBEN	2	0b	RW	Global DCB Enable. Loaded from the NVM and defines if DCB is enabled. Can be overridden by firmware. 0b = DCB disabled. 1b = DCB enabled. Used by firmware only.
VTEN	3	0b	RW	Global VT Enable. Loaded from the NVM and defines if the virtualization offloads are enabled. Can be overridden by firmware. 0b = VT disabled. 1b = VT enabled. Used by firmware only.



Field	Bit(s)	Init.	Access Type	Description
EVBEN	5	0b	RW	Global EVB Enable. Loaded from the NVM and defines if the port virtualizer structures (s-comp/port extender) offloads are enabled. Can be overridden by firmware. 0b = EVB disabled. 1b = EVB enabled. Used by firmware only.
Reserved	31:8	0x0	Reserved	Reserved.

10.2.1.1.2 General Port Configuration - PRTGEN_CNF[PRT] (0x000B8120 + 0x4*PRT, PRT=0...3)

This register contains configuration per Ethernet port loaded from NVM.

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
PORT_DIS	0	1b	RW	Port Disable. Defines if the Ethernet port is enabled from the NVM. Exception — Port 0 is always enabled and cannot be disabled from the NVM. 0b = Enabled. 1b = Disabled. Default: 0b = For port 0. 1b = For other ports.
ALLOW_PORT_DIS	1	0b	RW	Allow Port Disable. 0b = Asserting DEV_DIS_N has no effect on this port. 1b = Asserting DEV_DIS_N disables this port.
EMP_PORT_DIS	2	0b	RW	Set by the EMP to disable the Ethernet port. The NVM value for this bit should always be 0b (enabled). NVM should use the PORT_DIS bit to disable a port.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.1.3 General Port Configuration2 - PRTGEN_CNF2[PRT] (0x000B8160 + 0x4*PRT, PRT=0...3)

This register contains configuration per Ethernet port loaded from NVM.

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: PRT



Field	Bit(s)	Init.	Access Type	Description
ACTIVATE_PORT_LINK	0	0b	RW	When this field is set to 0b, the port's link is powered down. This field can be used by an application to disable the link until the software device driver is loaded and enables the link. Notes: 1. The PCIe functions associated with the port are not affected by the link loss. 2. Deactivating the link, using this configuration, is ignored when the interface is used for manageability. In order to implement this, hardware masks this configuration when the relevant port's PRTPM_GC.EMP_LINK_ON is set to 1b.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.1.4 General Port Status - PRTGEN_STATUS[PRT] (0x000B8100 + 0x4*PRT, PRT=0...3)

This register contains general port status information.

Reset Source: POR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
PORT_VALID	0	0b	RO	Port Valid. Denotes if the respective Ethernet port is enabled. 0b = Port is disabled. 1b = Port is enabled.
PORT_ACTIVE	1	0b	RO	Port Active. Denotes if the respective Ethernet port is active. 0b = Port is inactive and its respective PHY is in power down. 1b = Port is active and its respective PHY is powered up. PORT_ACTIVE is 1b when PORT_VALID = 1b and: 1. If the device is in D0 state, then port should be active. 2. If the software device driver activated the link (ACTIVATE_PORT_LINK = 1b) or the link is used for manageability. Else, when the device is in Dr state, the port should be active: 1. If link is used for manageability or WoL.
Reserved	31:2	0x0	Reserved	Reserved.

10.2.1.1.5 Global Reset Status - GLGEN_RSTAT (0x000B8188)

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DEVSTATE	1:0	0x0	RO	The device can be at one of the following states: 00b = Device active. 01b = Reset requested. 10b = Reset in progress. 11b = Reserved.



Field	Bit(s)	Init.	Access Type	Description
RESET_TYPE	3:2	0x0	RO	Reset type reflects one of the following resets that are/were in progress: 00b = POR. 01b = CORER. 10b = GLOBR. 11b = EMPR.
CORERCNT	5:4	0x0	RO	CORERCNT counts the number of initiated core resets since POR. The counter wraps around from 11b to 00b. Note that the counters count only the graceful resets asserted by the GLGEN_RTRIG register. It does not count resets initiated by the GLGEN_IMRTRIG register.
GLOBRCNT	7:6	0x0	RO	GLOBRCNT counts the number of initiated global resets since POR. The counter wraps around from 11b to 00b. Note that the counters count only the graceful resets asserted by the GLGEN_RTRIG register. It does not count resets initiated by the GLGEN_IMRTRIG register.
EMPRCNT	9:8	0x0	RO	EMPRCNT counts the number of initiated EMP resets since POR. The counter wraps around from 11b to 00b. Note that the counters count only the graceful resets asserted by the GLGEN_RTRIG register. It does not count resets initiated by the GLGEN_IMRTRIG register.
TIME_TO_RST	15:10	0x0	RO	The reset time is a down counter, loaded by GLRSTDEL following a GLOBR or CORER or EMPR. When GLOBRTIME reaches a zero value the actual reset is initiated.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.1.6 Global Reset Trigger - GLGEN_RTRIG (0x000B8190)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
CORER	0	0b	RW	Setting CORER triggers a graceful core reset flow.
GLOBR	1	0b	RW	Setting GLOBR triggers a graceful global reset flow.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.1.7 Global Reset Control - GLGEN_RSTCTL (0x000B8180)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GRSTDEL	5:0	0x8	RW	Global/core and EMP resets delay defined in 100 ms units. Setting GRSTDEL to 0x0 bypasses the delay counter.
Reserved	7:6	0x0	Reserved	Reserved.
ECC_RST_ENA	8	0b	RW	Graceful GLOBR reset on ECC in any memory other than the EMP memories. 1b = A detected ECC error generated a graceful GLOBR. 0b = A detected ECC error on these memories generates only an interrupt to the PFs.
Reserved	31:9	0x0	Reserved	Reserved.

10.2.1.1.8 VM Reset Trigger - VSIGEN_RTRIG[*VSI*] (0x00090000 + 0x4VSI*, *VSI*=0...383)**

Reset Source: CORER

NVM Load: No

Scope: *VSI*

Field	Bit(s)	Init.	Access Type	Description
VMSWR	0	0b	RW	VM software reset is initiated by setting the VMR bit. At completion of the reset flow, PF software clears this bit.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.1.9 VM Reset Status - VSIGEN_RSTAT[*VSI*] (0x00090800 + 0x4VSI*, *VSI*=0...383)**

Reset Source: CORER

NVM Load: No

Scope: *VSI*



Field	Bit(s)	Init.	Access Type	Description
VMRD	0	1b	RO	VM Software Reset Done Indication. This flag is cleared when the VM reset is initiated (by setting the <i>VMSWR</i> flag in the matched <i>VSIGEN_RTRIG</i> register or any stronger reset that affects the VM). It is set back to 1b when hardware completes its VM reset sequence.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.1.10 VF Reset Status - VPGEN_VFRSTAT[VF] (0x00091C00 + 0x4*VF, VF=0...127)

This register affects the VF but exposed only to the parent PF.

Reset Source: CORER

NVM Load: No

Scope: VP

Field	Bit(s)	Init.	Access Type	CFG Policy	Description
VFRD	0	1b	RO	UNDEFINED	VF Software Reset Done Indication. This flag is cleared when the VF reset is initiated (by setting the <i>VFSWR</i> flag in the matched <i>VPGEN_VFRTRIG</i> register or any stronger reset that affects the VF). It is set back to 1b when hardware completes its VF reset sequence.
Reserved	31:1	0x0	Reserved		Reserved.

10.2.1.1.11 VF Reset Status - VFGEN_RSTAT[VF] (0x00074400 + 0x4*VF, VF=0...127)

Reset Source: VFR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
VFR_STATE	1:0	0x0	RW	The VFR state defines the VFR reset progress as follows: 00b = VFR in progress. 01b = VFR completed. 10b, 11b = Reserved. This field is used to communicate the reset progress to the VF with no impact on hardware functionality.
Reserved	31:2	0x0	Reserved	Reserved.

10.2.1.1.12 VF Reset Trigger - VPGEN_VFRTRIG[VF] (0x00091800 + 0x4*VF, VF=0...127)

This register affects the VF but exposed only to the parent PF.

Reset Source: CORER

NVM Load: No

Scope: VP



Field	Bit(s)	Init.	Access Type	Description
VFSWR	0	0b	RW	VF software reset is done by the PF setting the <i>VFSWR</i> bit. At reset completion, the PF clears this bit.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.1.13 Global VF Level Reset Status - GLGEN_VFLRSTAT[n] (0x00092600 + 0x4*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VFLRE	31:0	0x0	RW1C	VFLR Status Indication. Bit 'm' in register 'n' reflects a VFLR event on VF index '32 x n + m'. While 'n' is the register index and 'm' is the bit index.

10.2.1.1.14 Global Clock Status - GLGEN_CLKSTAT (0x000B8184)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
CLKMODE	0	0b	RW	Clock Mode. Controls the operating mode of internal clocks: 0b = Performance mode - Device operates at its highest internal frequencies, independent of the Ethernet link speed. 1b = Power mode - Device adjusts its internal frequencies to match the speed of the Ethernet link.
Reserved	3:1	0x0	Reserved	Reserved.
U_CLK_SPEED	5:4	0x0	RO	Debug Feature. Represents the current speed of the upper clock (core_clk) as follows: 00b = 390.625 MHz. 01b = 195.3125 MHz. 10b = 97.65625 MHz. 11b = Reserved.
Reserved	7:6	0x0	Reserved	Reserved.
P0_CLK_SPEED	10:8	0x0	RO	Debug Feature. Represents the current speed of the Rx clock for MAC 0. Speeds are represented as follows: 3 bits per port (synchronized to the equivalent MAC clock): 000b = 100 MB. 001b = 1 GbE. 010b = 10 GbE. Other values are reserved.
Reserved	11	0b	Reserved	Reserved.



Field	Bit(s)	Init.	Access Type	Description
P1_CLK_SPEED	14:12	0x0	RO	Debug Feature. Represents the current speed of the Rx clock for MAC 1. Speeds are represented as follows: 3 bits per port (synchronized to the equivalent MAC clock): 000b = 100 MB. 001b = 1 GbE. 010b = 10 GbE. Other values are reserved.
Reserved	15	0b	Reserved	Reserved.
P2_CLK_SPEED	18:16	0x0	RO	Debug Feature. Represents the current speed of the Rx clock for MAC 2. Speeds are represented as follows: 3 bits per port (synchronized to the equivalent MAC clock): 000b = 100 MB. 001b = 1 GbE. 010b = 10 GbE. Other values are reserved.
Reserved	19	0b	Reserved	Reserved.
P3_CLK_SPEED	22:20	0x0	RO	Debug Feature. Represents the current speed of the Rx clock for MAC 3. Speeds are represented as follows: 3 bits per port (synchronized to the equivalent MAC clock): 000b = 100 MB. 001b = 1 GbE. 010b = 10 GbE. Other values are reserved.
Reserved	31:23	0x0	Reserved	Reserved.



10.2.1.1.15 Global GPIO Control - GLGEN_GPIO_CTL[n] (0x00088100 + 0x4*n, n=0...29)

General GPIO Control registers.

These registers control the mode of operation of general purpose I/O (GPIO) pins.

There are 30 GPIO pins out of which GPIOs 0 through 21 can be configured to various different modes and assigned to different ports while GPIOs 22 through 29 are loaded from the NVM as LEDs with fixed per port allocation.

There is one register per GPIO pin.

Register GLGEN_GPIO_CTL [n] controls GPIO pin.

This register is initialized only at LAN Power Good preserving the GPIO states across software and PCIe resets.

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRT_NUM	1:0	0x0	RW	Controls the port number associated with this GPIO pin. This field is valid when the un-assigned port number bit (PRT_NUM_NA) is set to 0b. If a port number is not assigned then this GPIO pin is a global resource not associated with any specific port.
Reserved	2	0b	Reserved	Reserved.
PRT_NUM_NA	3	0b	RW	0b = SDP output might be assigned to High Z when a respective port is in power down or if global reset is active on the port specified in the PRT_NUM field. 1b = SDP value controlled by GLGEN_GPIO_SET register regardless of port state. This bit effects hardware only when the respective GPIO is configured to SDP mode.
PIN_DIR	4	0b	RW	Controls whether this GPIO pin is configured as an input or output (0b = input, 1b = output). This bit is not affected by software or system reset, only by initial power on or direct software writes. Note: When GPIO functionality is set to LED, the pin direction is set implicitly to output regardless of this field's value.
TRI_CTL	5	0b	RW	Tristate Control. When this GPIO pin is configured as an output, this field controls whether the pin is driven to tristate or driven high (0b = tristate, 1b = driven high) when SDP_DATA is set to 1b. The pin is driven to active low when the SDP_DATA is set to 0b. This bit is not affected by software or system reset, only by initial power on or direct software writes. SDP_DATA is set through the GLGEN_GPIO_SET register. Note: This field is not used when GPIO is set to LED functionality.
OUT_CTL	6	0b	RW	Controls the output state during reset or D3 power state when no WoL or no manageability. When this bit is set to 0b the output transitions to tristate during reset or D3 power state. When this bit is set to 1b, the output is driven high during reset or D3 power state. This bit is ignored when this GPIO pin is configured as an input. This bit is not affected by software or system reset, only by initial power on or direct software writes. Note: This field is not used when GPIO functionality is set to LED.



Field	Bit(s)	Init.	Access Type	Description
PIN_FUNC	9:7	0x0	RW	This field controls the operation mode (or functionality) for this GPIO pin. 000b = SDP (software definable input or output). 001b = LED (drives external LEDs). 010b = RoL (reset on LAN). 011b = Timesync 0. 100b = Timesync 1. 101b = Reserved. 110b = Firmware LED 111b = PHY Diagnostics. Used to connect to internal PHYs indications. When this pin is configure in SDP mode, the output value is set through GLGEN_GPIO_SET register and the value of the GPIO pin is read through GLGEN_GPIO_STAT register.
LED_INVRT	10	0b	RW	This field specifies the polarity/inversion of the LED source prior to output or blink control. By default, the output drives the cathode of the LED so when the LED output is 0b the LED is on. 0b = LED output is active low. 1b = LED output is active high. The LED_INVRT setting is meaningful only if the PIN_FUNC equals to LED.
LED_BLINK	11	0b	RW	This field specifies whether or not to apply blink logic to the (inverted) LED control source prior to the LED output. 0b = Do not blink LED output. 1b = Blink LED output. The LED_BLINK setting is meaningful only if the PIN_FUNC equals to LED.
LED_MODE	16:12	0x0	RW	This field specifies the control source for the LED output. LED modes are described in detail in the LED section. The LED_MODE setting is meaningful only if the PIN_FUNC equals to LED.
INT_MODE	18:17	0x0	RW	This field selects the interrupt mode for this GPIO pin. 00b = No interrupt. 01b = Interrupt on rising edge. 10b = Interrupt on falling edge. 11b = Interrupt on any transition. Set this field to generate interrupts only when PIN_FUNC is equal to SDP.
OUT_DEFAULT	19	0b	RW	Default value driven on this GPIO pin when configured as a SDP output. The value is ignored when GPIO pin is configured as a SDP input or when PIN_FUNC is not SDP. 0b = Output driven low. 1b = Output driven high.



Field	Bit(s)	Init.	Access Type	Description
PHY_PIN_NAME	25:20	0x0	RW	This field is used to indicate to firmware the name and functionality for which this GPIO pin is being used when connecting to an external PHY/module control/status pin. 0x3F = SDP is not used for PHY/module connectivity. 0x0 to 0x7 = Control/status pin names when connecting to an SFP+ module. 0x0 = Rx_LOS. 0x1 = Mod_ABS. 0x2 = RS0/RS1. 0x3 = TxDisable. 0x4 = TxFault. 0x8 to 0xF = Reserved. 0x10 to 0x17 = Control status pin names when connecting to 10GBASE-T PHYs. 0x10 = PhyInt. 0x11 = PhyRst. 0x12 = TxDisable. All other values are reserved.
PRT_BIT_MAP	29:26	0x0	RW	Each bit in this field indicates the port number of the external PHY or module associated with this GPIO. When a GPIO is not associated with any port then bits 0-3 are all zero. This field is meaningful only when the PHY_PIN_NAME field is set to one of the following: 0x8 = IntL. 0x9 = ResetL. 0xa = ModPresL. 0xb = LPMoDe. 0xc = ModSelL. In this case, PRT_NUM_NA must be set to 0x1 and the PRT_NUM is meaningless. If the PHY_PIN_NAME field is set differently than previous, PRT BIT MAP is meaningless and must be set to zero.
Reserved	31:30	0x0	Reserved	Reserved.

10.2.1.1.16 Global LED Control - GLGEN_LED_CTL (0x00088178)

Global LED Control.

Configures global LED blink mode.

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GLOBAL_BLINK_MODE	0	0b	RW	Global Blink Mode. This field specifies the blink mode of all LEDs. 0b = Blink at 200 ms on and 200 ms off. 1b = Blink at 83 ms on and 83 ms off.
Reserved	31:1	0x0	Reserved	Reserved.



10.2.1.1.17 Global GPIO Status - GLGEN_GPIO_STAT (0x0008817C)

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GPIO_VALUE	29:0	0x0	RO	Each bit 'n' in this field reflects the value of GPIO pin (either input or output). For example, bit 0 reflects the value on GPIO0 pin (1b is high and 0b is low) and bit 1 indicates the value of GPIO1 pin and so on (see Section 3.5).
Reserved	31:30	0x0	Reserved	Reserved.

10.2.1.1.18 Global GPIO Transition Status - GLGEN_GPIO_TRANSIT (0x00088180)

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GPIO_TRANSITION	29:0	0x0	RW1C	These register bits are used to latch any transition (low-to-high or high-to-low) on the GPIO pins (either input or output) since the last time the register bits were cleared. Bit 0 reflects the transition status of GPIO0 pin and bit 1 indicates the transition status of GPIO1 pin and so on (see Section 3.5). The PF is expected to clear those flags that it cares about by writing 1b to the corresponding bit position. The PF is expected to clear the flags only for the GPIO pins that it owns.
Reserved	31:30	0x0	Reserved	Reserved.

10.2.1.1.19 Global GPIO Set - GLGEN_GPIO_SET (0x00088184)

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GPIO_INDX	4:0	0x0	RW	Define the GPIO pin number that is driven to SDP_DATA value. The GPIO_INDX can be set to any value between 0 and 29 decimal (00000b to 11001b). See Section 3.5 for more detail. Note that software is permitted to drive only GPIO pins that are defined as output SDPs by the GLGEN_GPIO_CTL registers. Also note that any PFs or EMP can access all GPIOs. It is the PF software responsibility to control only those GPIOs that the PF owns.
SDP_DATA	5	0b	RW	The value in this field is driven to a GPIO pin pointed by the index GPIO_INDX. 0b = Driven low. 1b = Driven high.
DRIVE_SDP	6	0b	RW	Set this flag to 1b to drive the SDP_DATA to GPIO pin pointed by the index GPIO_INDX.
Reserved	31:7	0x0	Reserved	Reserved.



10.2.1.1.20 Global MDIO or I²C Select - GLGEN_MDIO_I2C_SEL[n] (0x000881C0 + 0x4*n, n=0..3)

This register is used to select between the MDIO interface or I²C interface on the MDIO_n_SDAn/ MDC_n_SCL_n pins. Each pair of these pins can be independently configured for MDIO or I²C interface protocol by using GLGEN_MDIO_I2C_SEL[n] registers.

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MDIO_I2C_SEL	0	0b	RW	The MDIO_I2C_SEL bit is used to select between the MDIO interface or I ² C interface over the MDIO _n _SDAn/MDC _n _SCL _n pins, where n=0..3. 0b = MDIO interface is selected. 1b = I ² C interface is selected.
PHY_PORT_NUM	4:1	0x0	RW	Each bit in this field indicates the port number of the external PHY or module associated with this GPIO. This field is a bitmap so more than one bit might be set to 1 at any given time. Bit 0 = Port 0 PHY/module. Bit 1 = Port 1 PHY/module. Bit 2 = Port 2 PHY/module. Bit 3 = Port 3 PHY/module. Notes: 1. When a GPIO is not associated with any port then bits 0-3 are all zero. 2. For sharing a single I ² C/MDIO between four ports I ² C-0/MDIO-0 should be used. 3. For sharing a single I ² C/MDIO between two ports I ² C-0/MDIO-0 should be used for ports 0,1 and I ² C-1/MDIO-1 should be used for ports 2,3.
PHY0_ADDRESS	9:5	0x0	RW	If this interface is used for connecting to PHY0 and is the address of the PHY device to use. This field is used by firmware.
PHY1_ADDRESS	14:10	0x0	RW	If this interface is used for connecting to PHY1 and is the address of the PHY device to use. This field is used by firmware.
PHY2_ADDRESS	19:15	0x0	RW	If this interface is used for connecting to PHY2 and is the address of the PHY device to use. This field is used by firmware.
PHY3_ADDRESS	24:20	0x0	RW	If this interface is used for connecting to PHY3 and is the address of the PHY device to use. This field is used by firmware.
MDIO_IF_MODE	28:25	0x0	RW	When interface mode is set to MDC/MDIO, this field indicates to firmware the frame structure to use. This field contains a bit per port for scenarios where more than one PHY device is connected through a single interface. Each bit indicates the following options: 1b = Use IEEE Clause 22 Frame Structure. 0b = Use IEEE Clause 45 Frame Structure. Bit0 = Port0 MDIO_IF_MODE. Bit1 = Port1 MDIO_IF_MODE. Bit2 = Port2 MDIO_IF_MODE. Bit3 = Port3 MDIO_IF_MODE.
Reserved	30:29	0x0	Reserved	Reserved.



10.2.1.1.21 MDIO Control - GLGEN_MDIO_CTRL[n] (0x000881D0 + 0x4*n, n=0..3)

This register is used to control the MDC clock speed and mode when the MDION_SDAn/MDCn_SCLn pins are configured for use as a MDIO interface. Each pair of these pins are independently configured as a MDIO or I²C interface through the GLGEN_MDIO_I2C_SEL[n] registers, where n=0..3.

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LEGACY_RSVD2	16:0	0x2FFB	RW	Reserved. Do not modify this value.
CONTMDC	17	0b	RW	Continuous MDC. Turn off MDC between MDIO packets. 1b = Continuous MDC. 0b = MDC off between packets (default).
LEGACY_RSVD1	28:18	0x200	RW	Reserved. Do not modify this value.
LEGACY_RSVD0	31:29	0x0	RO	Reserved. Do not modify this value.

10.2.1.1.22 MDI Single Command and Address - GLGEN_MSCA[n] (0x0008818C + 0x4*n, n=0..3)

This register is used for writing the command and address to initiate read/write access over the MDIO interface. This register is used when the MDION_SDAn/MDCn_SCLn pins are configured for MDIO operation through the GLGEN_MDIO_I2C_SEL[n] register.

Note: The access type in the PF and VF spaces is determined individually per field. The values are contained in the table that describes the fields in the internal space.

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MDIADD	15:0	0x0000	RW	MDI Address. Address used for new protocol MDI accesses (default = 0000b).
DEVADD	20:16	0x00	RW	DEVICE TYPE/REG ADDR. Five bits representing either device type if STCODE = 00b or register address if STCODE = 01b.
PHYADD	25:21	0x00	RW	PHY Address. The address of the external device.
OPCODE	27:26	0x00	RW	Opcode. Two bits identifying operation to be performed (default = 00b). 00b = Address cycle (new protocol only). 01b = Write operation. 10b = Read, increment address (new protocol only). 11b = Read operation.
STCODE	29:28	0x01	RW	ST Code. Two bits identifying start of frame and old or new protocol (default = 01b). 00b = New protocol. 01b = Old protocol. 1x = Illegal.



Field	Bit(s)	Init.	Access Type	Description
MDICMD	30	0b	RW1C	MDI Command. Perform the MDI operation in this register; cleared when done. 1b = Perform operation; operation in progress. 0b = MDI ready; operation complete (default).
MDIINPROGEN	31	0b	RW	MDI In Progress Enable. Generate MDI in progress when operation completes. 1b = MDI in progress enable. 0b = MDI ready disable (default).

10.2.1.1.23 MDI Single Read and Write Data - GLGEN_MSRWD[n] (0x0008819C + 0x4*n, n=0..3)

This register is used for reading and writing data to and from the MDIO interface. This register is used when the MDION_SDAn/MDCn_SCLn pins are configured for MDIO operation through the GLGEN_MDIO_I2C_SEL[n] register.

Note: The access type in the PF and VF spaces is determined individually per field. The values are contained in the table that describes the fields in the internal space.

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MDIWRDATA	15:0	0x0	RW	MDI Write Data. Write data For MDI writes to the external device.
MDIRDDATA	31:16	0x0	RO	MDI Read Data. Read data from the external device.

10.2.1.1.24 I²C Command - GLGEN_I2CCMD[n] (0x000881E0 + 0x4*n, n=0..3)

This register is used to read or write to the configuration registers over the I²C interface when the MDION_SDAn/MDCn_SCLn pins are configured for I²C operation. Each pair of these pins are independently configured as a MDIO or I²C interface through the GLGEN_MDIO_I2C_SEL[n] registers, where n=0..3.

Note: The access type in the PF and VF spaces is determined individually per field. The values are contained in the table that describes the fields in the internal space.

Reset Source: POR

NVM Load: No, Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DATA	15:0	0x0	RW	I ² C Data. For a write command, firmware/software places the data bits in this field and hardware shifts them out to the I ² C bus. For a read command, hardware reads the bits serially from the I ² C bus and places the data in this field so firmware/software can fetch the data from this location. Note: This field is read in byte order and not in word order. Ordering can be changed by using the GLGEN_I2CPARAMS.I2C_DATA_ORDER bit.



Field	Bit(s)	Init.	Access Type	Description
REGADD	23:16	0x0	RW	I ² C Register Address. For example, register 0, 1, 2... 255.
PHYADD	26:24	0x0	RW	Device Address Bits 2:0. The actual address used is 1010b (PHYADD[2:0]).
OP	27	0b	RW	Opcode. 0b = I ² C write. 1b = I ² C read.
RESET	28	0b	RW1C	Reset Sequence. If set, sends a reset sequence before the actual read or write. This bit is self clearing. A reset sequence is defined as nine consecutive stop conditions.
R	29	0b	RW	Ready Bit. Indicates a read or write operation on the I ² C interface has completed. Set to 1b by hardware at the end of the I ² C transaction. Reset by a firmware/software write of a new command to this register.
Reserved	30	0b	Reserved	Reserved.
E	31	0b	RW	Error. This bit set is to 1b by hardware when it fails to complete an I ² C read. Reset by a firmware/software write of a new command. Note: This bit is valid only when Ready bit R is set to 1b by hardware.

10.2.1.1.25 I2C Parameters - GLGEN_I2CPARAMS[n] (0x000881AC + 0x4*n, n=0...3)

This register is used to set the parameters for the I²C access to the 2-wire management interface and to enable bit banging access to the I²C interface. This register is used when the MDIO_n_SDAn/ MDC_n_SCL_n pins are configured for I²C interface operation through the GLGEN_MDIO_I2C_SEL[n] register.

Note: The access type in the PF and VF spaces is determined individually per field. The values are contained in the table that describes the fields in the internal space.

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
WRITE_TIME	4:0	0x6	RW	Write Time. Defines the delay between a write access and the next access. The value is in ms. A value of 0b is not valid.
READ_TIME	7:5	0x2	RW	Read Time. Defines the delay between a read access and the next access. The value is in ms. A value of 0b is not valid.
I2CBB_EN	8	0b	RW	I ² C Bit Bang Enable. If set, the I2C_CLK and I2C_DATA lines are controlled via the CLK, DATA and DATA_OE_N fields of this register. Otherwise, they are controlled by the hardware machine activated via the GLGEN_I2CCMD[n] register.



Field	Bit(s)	Init.	Access Type	Description
CLK	9	1b	RW	I ² C CLK out value used to drive the value of I2C_CLK (SCL output to PAD). While in bit bang mode, controls the value driven on the I ² C clock pad MDCn_SCLn.
DATA_OUT	10	1b	RW	I ² C DATA_OUT value used to drive the value of I ² C data (SDA output to PAD). While in bit bang mode and when the DATA_OE_N field is zero, controls the value driven on the I ² C data pad MDION_SDAn.
DATA_OE_N	11	0b	RW	I ² C DATA_OE_N. While in bit bang mode, controls the direction of the I2C_DATA pad MDION_SDAn. 0b = Pad is output. 1b = Pad is input.
DATA_IN	12	0b	RO	I ² C DATA_IN. Provides the value of I2C_DATA (SDA input from external PAD). This bit is RO. Reflects the value of the I2C_DATA pad MDION_SDAn. While in bit bang mode when the DATA_OE_N field is zero, this field reflects the value set in the DATA_OUT field.
CLK_OE_N	13	0b	RW	I ² C Clock Output Enable. While in bit bang mode, controls the direction of the I2C_CLK pad MDCn_SCLn. 0b = Pad is output. 1b = Pad is input.
CLK_IN	14	0b	RO	I ² C Clock In Value (SCL input from external PAD). This bit is RO. Reflects the value of the I ² C CLK pad MDCn_SCLn. While in bit bang mode when the CLK_OE_N field is zero, this field reflects the value set in the CLK_OUT field.
CLK_STRETCH_DIS	15	0b	RW	I ² C Clock Stretch Disable. 0b = Enable slave clock stretching support in I ² C access. 1b = Disable clock stretching support in I ² C access.
Reserved	30:16	0x0	Reserved	Reserved.

10.2.1.1.26 Global Device Timer - GLVFGEN_TIMER (0x000881BC)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GTIME	31:0	0x0	RW	GTIME is a free running timer fed by a 1 μ s clock.

10.2.1.1.27 PF Control - PFGEN_CTRL[PF] (0x00092400 + 0x4*PF, PF=0...15)

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PFSWR	0	0b	RW	PF software reset is done by setting the PFSWR bit. At reset completion, hardware clears this bit.



Field	Bit(s)	Init.	Access Type	Description
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.1.28 PF State - PFGEN_STATE[PF] (0x00088000 + 0x4*PF, PF=0...15)

This register defines to software the main characteristics of the PF. It does not have any direct impact on device functionality.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PFLINKEN	2	0b	RW	PF Link Enable. 0b = The PF driver is disabled. The software device driver must at minimum report it does not have link. 1b = The PF driver is enabled.
PFSCEN	3	0b	RW	PF iSCSI Enable. 0b = iSCSI should not be used for this function. 1b = iSCSI can be used for this function together with LAN functionality.
Reserved	31:4	0x0	Reserved	Reserved.

10.2.1.1.29 PF Driver Unload - PFGEN_DRUN[PF] (0x00092500 + 0x4*PF, PF=0...15)

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
DRVUNLD	0	0b	RW	PF sets this bit to indicate its driver is unloading.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.1.30 LAN Port Number - PFGEN_PORTNUM[PF] (0x001C0480 + 0x4*PF, PF=0...15)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PORT_NUM	1:0	0x0	RW	Port Number. Indicates the LAN port connected to this function. 00b = Port 0. 01b = Port 1. 10b = Port 2. 11b = Port 3.
Reserved	31:2	0x0	Reserved	Reserved.



10.2.1.1.31 Firmware Status Register - GL_FWSTS (0x00083048)

This register reports the status of the EMP.

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FWS0B	7:0	0x0	RO	Firmware Status 0 Byte. This bit is RO through the host interface.
Reserved	8	0b	Reserved	Reserved.
FWRI	9	1b	RW1C	Firmware Reset Indication. Set when a firmware reset is asserted. Cleared when the host writes a 1b to it. Writing a 0b to this bit does not change its value. Note: This bit is also set after LAN_PWR_GOOD and is RO through the AUX interface.
Reserved	15:10	0x0	Reserved	Reserved.
FWS1B	23:16	0x0	RO	Firmware Status 1 Byte. This bit is RO through the host interface. 0x30 = STATUS_RECOVERY_MODE_CORER. 0x31 = STATUS_RECOVERY_MODE_GLOBR. 0x32 = STATUS_RECOVERY_MODE_TRANSITION. 0x33 = STATUS_RECOVERY_MODE_NVM. Others: Reserved.
Reserved	31:24	0x0	Reserved	Reserved.

10.2.1.2 PCIe Registers

This category contains registers for PCIe configuration and control.

10.2.1.2.1 PCIe PM - PFPCI_PM[PF] (0x000BE300 + 0x4*PF, PF=0...15)

Reset Source: POR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	CFG Policy	Description
PME_EN	0	0b	RW	UNDEFINED	PME Enable. This read/write bit is used by the software device driver to generate a PME event without writing to the PMCSR in the PCIe configuration space. Note: The internal PME enablement is a logic OR function of the following: PME_EN flag in the PMCSR, PME_EN flag in the PFPCI_PM_CSR and APME flag in the PFPM_APM_CSR. The bit is reset on STRST (Sticky Reset): bit is reset only on power-on reset (LAN_PWR_GOOD). When AUX_PWR = 0b, this bit is also reset when de-asserting PE_RST_N.
Reserved	31:1	0x0	Reserved		Reserved.

**10.2.1.2.2 PCIe Vendor ID - GLPCI_VENDORID (0x000BE518)**

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VENDORID	15:0	0x8086	RW	Contains the vendor ID exposed in offset 0x0 in the config space of all functions. A value of 0xFFFF is ignored.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.2.3 PF PCIe Subsystem ID - PFPCI_SUBSYSID[PF] (0x000BE100 + 0x4*PF, PF=0...15)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PF_SUBSYS_ID	15:0	0x0	RW	Subsystem ID for this PF.
VF_SUBSYS_ID	31:16	0x0	RW	Subsystem ID for VFs of this PF.

10.2.1.2.4 PCI Function Count - GLGEN_PCIFCNCNT (0x001C0AB4)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PCIPFCNT	4:0	0x1	RW	Reports the function number of the highest PCI physical function plus 1 as it is loaded from the NVM. For example, if the NVM enables 2 functions 0 and 7, the value is 8 (7+1). Used to partition the interrupt vectors among supported PFs.
Reserved	15:5	0x0	Reserved	Reserved.
PCIVFCNT	23:16	0x0	RW	Reports the function number of highest PCI virtual function plus 1 for all PFs combined, as it is loaded from the NVM. Used to partition the interrupt vectors among supported VFs.
Reserved	31:24	0x0	Reserved	Reserved.



10.2.1.2.5 PCIe* Global Config 2 - GLPCI_CNF2 (0x000BE494)

This register contains global status fields of PCIe configuration.

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RO_DIS	0	0b	RW	Relaxed Ordering Disable. When set to 1b, the device does not request any relaxed ordering transactions. When this bit is cleared, relaxed ordering is specified per request type.
CACHELINE_SIZE	1	0b	RW	Cache Line Size. Determines the system cache line size. 0b = 64 bytes. 1b = 128 bytes. This field is loaded from the NVM.
MSI_X_PF_N	12:2	0x80	RW	MSI_X PF Table Size. System software reads this field to determine the MSI-X table size N, which is encoded as N-1. This field is loaded from the NVM MSI_X _N_PF field and reflects the same field in the PCIe MSI-X configuration (Message Control register).
MSI_X_VF_N	23:13	0x04	RW	MSI_X VF Table Size. System software reads this field to determine the MSI-X table size N for VFs, which is encoded as N-1. This field is loaded from the NVM MSI_X _N_VF field and reflects the same field in the PCIe MSI-X configuration (VF MSI-X Control register).
Reserved	31:24	0x0	Reserved	Reserved.



10.2.1.2.6 PCI BAR Control - GLPCI_LBARCTRL (0x000BE484)

Reset Source: POR

NVM Load: Forbidden Load

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PREFBAR	0	1b	RW	Prefetchable bit indication in the memory BAR and MSI-X BAR (should be set when 64-bit BARs are used). 0b = BARs are marked as non-prefetchable. 1b = BARs are marked as prefetchable.
BAR32	1	0b	RW	BAR 32-bit Enable. 0b = 64-bit BAR addressing mode is selected. 1b = 32-bit BARs are enabled.
Reserved	2	0b	Reserved	Reserved.
FLASH_EXPOSE	3	1b	RW	When set, Flash memory is accessible through the memory BAR.
FL_SIZE	8:6	0x7	RW	This field indicates the size of the external Flash as = 64 KB x (2 ** FL_SIZE). The following values are supported: 101b = 2 MB. 110b = 4 MB. 111b = 8 MB. Else = Reserved.
Reserved	9	0b	Reserved	Reserved.
EXROM_SIZE	13:11	0x3	RW	This field indicates the size of the expansion ROM BAR as = 64 KB x (2 ** EXROM_SIZE). 000b = 64 KB size. 111b = 8 MB size. Default value is 512 KB.
Reserved	31:14	0x0	Reserved	Reserved.

10.2.1.2.7 PCIe* Global Config - GLPCI_CNF (0x000BE4C0)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
Reserved	0	0b	Reserved	
WAKE_PIN_EN	2	0b	RW	When set to 1b, enables the use of the WAKE pin for a PME event in all power states.
Reserved	31:3	0x0	Reserved	Reserved.



10.2.1.2.8 PCIe Capabilities Support - GLPCI_CAPSUP (0x000BE4A8)

Determines PCIe capabilities supported by the device.

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PCIE_VER	0	1b	RW	Determines the PCIe capability version. 0b = Capability version: 0x1. 1b = Capability version: 0x2.
Reserved	1	0b	Reserved	Reserved.
TPH_EN	3	1b	RW	A value of 1b indicates support for the PCIe TPH requester capability.
ARI_EN	4	1b	RW	A value of 1b indicates support for PCIe ARI capability.
IOV_EN	5	1b	RW	A value of 1b indicates support for PCIe SR-IOV capability.
ACS_EN	6	1b	RW	A value of 1b indicates support for PCIe ACS capability.
SEC_EN	7	1b	RW	A value of 1b indicates support for the secondary PCI express extended capability.
Reserved	15:8	0x0	Reserved	Reserved.
ECRC_GEN_EN	16	1b	RW	Loaded into the ECRC. Generation capable bit of the PCIe configuration registers.
ECRC_CHK_EN	17	1b	RW	Loaded into the <i>ECRC Check</i> capable bit of the PCIe configuration registers.
IDO_EN	18	1b	RW	Enables ID-based ordering (IDO).
MSI_MASK	19	1b	RW	MSI Per-vector Masking Setting. This bit is loaded to the masking bit (bit 8) in the message control of the MSI configuration capability structure.
CSR_CONF_EN	20	1b	RW	Enables access to CSRs via the PCI configuration space. See the section on Configuration Access to Internal Registers and Memories.
Reserved	29:21	0x0	Reserved	Reserved.
LOAD_SUBSYS_ID	30	0b	RW	Load Subsystem IDs. When set to 1b, indicates that the device loads its PCIe sub-system ID and sub-system vendor ID from the NVM.
LOAD_DEV_ID	31	0b	RW	Load Device ID. When set to 1b, indicates that the device loads its PCI device ID's from the NVM.



10.2.1.2.9 PCIe Link Capabilities - GLPCI_LINKCAP (0x000BE4AC)

Determines PCIe Link capabilities supported by the device.

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LINK_SPEEDS_VECTOR	5:0	0x00	RW	Supported Link Speeds Vector. Loaded to the Link Capabilities 2 register in the PCIe capability values: Bit 0 = 5.0 GT/s. Bit 1 = 8.0 GT/s. Bits 5:2 = Reserved. Note: 2.5 GT/s is always supported.
MAX_PAYLOAD	8:6	0x4	RW	Max Payload Size Supported. Loaded to the PCIe Device Capabilities register. Supported values are 000b-to-100b (128 bytes-to-2 KB). Otherwise, keep the hardware default.
MAX_LINK_WIDTH	12:9	0x07	RW	Max Link Width. Loaded to the PCIe Link Capabilities register. Values: 0001b = Limit max link width to x1. 0011b = Limit max link width to x4. 0100b = Limit max link width to x8. 0111b = Do not limit max link width. Negotiate to the max width supported by the link. Else = Reserved.
Reserved	31:13	0x0	Reserved	Reserved.

10.2.1.2.10 PCIe Capabilities Control - GLPCI_CAPCTRL (0x000BE4A4)

Determines PCIe capabilities supported by the device and that software is allowed to enable or disable.

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VPD_EN	0	0b	RW	0b = PCIe VPD capability is not present and is not exposed. 1b = PCIe VPD capability is present and exposed.
Reserved	31:1	0x0	Reserved	Reserved.



10.2.1.2.11 TPH Control Register - GLTPH_CTRL (0x000BE480)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
Reserved	7:0	0x0	Reserved	Reserved.
DESC_PH	10:9	0x0	RW	Descriptor PH. Defines the <i>PH</i> field used when a TPH hint is given for descriptor associated traffic (descriptor fetch, descriptor write back or head write back). The default value should be kept for regular operation.
DATA_PH	12:11	0x2	RW	Data PH. Defines the <i>PH</i> field used when a TPH hint is given for data associated traffic (Tx data read, Rx data write). The default value should be kept for regular operation
Reserved	31:14	0x0	Reserved	Reserved.

10.2.1.2.12 PCIe Serial Number MAC Address Low - GLPCI_SERL (0x000BE498)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
SER_NUM_L	31:0	0x0	RW	The low Dword of the Ethernet MAC address used to generate the PCIe serial number. The register contents is loaded from the NVM. The location in the NVM is pointed by the 4th item in the Auto-Generated Pointers module. It is a per-device manufacturing value that represents the entire device. It can be set identical to the concatenated [PFPM_SAL1 PFPM_SALO] words of the EMP Settings module.



10.2.1.2.13 PCIe Serial Number MAC Address High - GLPCI_SERH (0x000BE49C)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
SER_NUM_H	15:0	0x0	RW	The high word of the Ethernet MAC address used to generate the PCIe serial number. The register contents is loaded from NVM. The location in NVM is pointed by the 5th item in the Auto-Generated Pointers module. It is a per device manufacturing value which represents the whole device. It can be set identical to the concatenated [PFPM_SAH1 PFPM_SAH0] words of the EMP Settings module.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.2.14 PCIe Storage Class - PFPCI_CLASS[PF] (0x000BE400 + 0x4*PF, PF=0...15)

Contains per-PF configuration loaded from NVM.

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
STORAGE_CLASS	0	0b	RW	0b = The class code of this function is set to 0x020000 (LAN). 1b = The class code of this function is set to 0x010000 (SCSI).
RESERVED_1	1	0b	RW	Reserved.
PF_IS_LAN	2	0b	RW	0b = SAN function. 1b = LAN function.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.2.15 PCIe PF Configuration - PFPCI_CNF[PF] (0x000BE000 + 0x4*PF, PF=0...15)

Contains per-PF configuration loaded from NVM.

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
Reserved	1:0	0x0	Reserved	Reserved.
MSI_EN	2	1b	RW	Enables the MSI capability structure for this PCI function. 0b = MSI is disabled. 1b = MSI is enabled.



Field	Bit(s)	Init.	Access Type	Description
EXROM_DIS	3	0b	RW	Expansion ROM Disable. 0b = The Expansion ROM BAR in the PCI configuration space is enabled. 1b = The Expansion ROM BAR in the PCI configuration space is disabled.
IO_BAR	4	0b	RW	I/O BAR Support. 0b = I/O BAR is not supported. 1b = I/O BAR is supported.
INT_PIN	6:5	0x0	RW	Controls the value advertised in the <i>Interrupt Pin</i> field of the PCI configuration header for this function. Values of 00b, 01b, 10b and 11b correspond to INTA#, INTB#, INTC# and INTD#, respectively. The value advertised in the PCI configuration header is the value loaded from NVM + 1.
Reserved	31:7	0x0	Reserved	Reserved.

10.2.1.2.16 PCIe Functions Configuration - PFPCI_FUNC[PF] (0x000BE200 + 0x4*PF, PF=0...15)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FUNC_DIS	0	1b	RW	Function Disable. Defines if the PCI function is enabled from the NVM. Exception: This bit is RO for PF0. It is always enabled and cannot be disabled from the NVM. 0b = Enabled. 1b = Disabled. Default: 0b for PF0. 1b for other functions.
ALLOW_FUNC_DIS	1	0b	RW	Allow Function Disable. 0b = Asserting PCI_DIS_N has no effect on this PCI function. 1b = Asserting PCI_DIS_N disables this PCI function.
DIS_FUNC_ON_PORT_DIS	2	0b	RW	Defines whether this PF is disabled when the DEV_DIS_N pin is asserted. 0b = Asserting DEV_DIS_N has no effect on this PCI function. 1b = Asserting DEV_DIS_N disables this PCI function.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.2.17 PCIe Functions Configuration 2 - PFPCI_FUNC2[PF] (0x000BE180 + 0x4*PF, PF=0...15)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: PF



Field	Bit(s)	Init.	Access Type	Description
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.2.18 PCIe VF Capabilities Support - GLPCI_VFSUP (0x000BE4B8)

Reset Source: PCIR.

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VF_PREFETCH	0	1b	RW	VF Prefetchable. 0b = IOV memory BAR and MSI-X BAR are declared as non-prefetchable. 1b = IOV memory BAR and MSI-X BAR are declared as prefetchable.
VR_BAR_TYPE	1	1b	RW	VF BAR Type. 0b = VF BARs advertise 32-bit size. 1b = VF BARs advertise 64-bit size.
Reserved	31:2	0x0	Reserved	Reserved.

10.2.1.2.19 PCIe Default Revision ID - GLPCI_DREVID (0x0009C480)

Reset Source: PCIR.

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DEFAULT_REVID	7:0	0x01	RO	Mirroring of default Rev ID prior to an NVM load.
Reserved	31:8	0x00	Reserved	Reserved.

10.2.1.2.20 Function Active and Power State - PFPCI_FACTPS[PF] (0x0009C180 + 0x4*PF, PF=0...15)

This register provides different indications about the function status.

Reset Source: FLR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FUNC_POWER_STATE	1:0	0x00	RO	Power state indication of the function. 00b = Dr. 01b = D3. 10b = D0a. 11b = D0u.
Reserved	2	0b	Reserved	Reserved.



Field	Bit(s)	Init.	Access Type	Description
FUNC_AUX_EN	3	0b	RO	Function Aux Enable. Reflects the <i>Auxiliary Power PM</i> Enable bit from the PCI configuration space.
Reserved	31:4	0x00	Reserved	Reserved.

10.2.1.2.21 PFPCI_STATUS1 - PFPCI_STATUS1[PF] (0x000BE280 + 0x4*PF, PF=0...15)

Reset Source: POR

NVM Load: No, Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FUNC_VALID	0	0b	RO	Function Valid. 0b = Function is disabled. 1b = Function is enabled. Note: This bit is valid to firmware even when the function is disabled.
Reserved	31:1	0x0	Reserved	

10.2.1.2.22 Function Requester ID Information Register - PF_FUNC_RID[PF] (0x0009C000 + 0x4*PF, PF=0...15)

Reset Source: PCIR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FUNCTION_NUMBER	2:0	0x0	RO	Function number assigned to the function based on BIOS/OS enumeration.
DEVICE_NUMBER	7:3	0x0	RO	No-ARI Mode. Device number assigned to the function based on BIOS/OS enumeration. ARI mode: upper 5 bits of the 8-bit function number.
BUS_NUMBER	15:8	0x0	RO	Bus number assigned to the function based on BIOS/OS enumeration.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.2.23 PCIe PM Support - GLPCI_PMSUP (0x000BE4B0)

This register contains parameters that define PCIe power management support.

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
Reserved	1:0	0x3	RW	Reserved.
LOS_EXIT_LAT	4:2	0x5	RW	LOs Exit Latency. Loaded to <i>LOs Exit Latency</i> field in the PCIe Link Capabilities register.



Field	Bit(s)	Init.	Access Type	Description
L1_EXIT_LAT	7:5	0x4	RW	L1 Exit Latency. Loaded to <i>L1 Exit Latency</i> field in the PCIe Link Capabilities register.
L0S_ACC_LAT	10:8	0x3	RW	Loaded to the Endpoint <i>L0s Acceptable Latency</i> field in the PCIe Device Capabilities register.
L1_ACC_LAT	13:11	0x6	RW	Loaded to the Endpoint <i>L1 Acceptable Latency</i> field in the PCIe Device Capabilities register.
SLOT_CLK	14	1b	RW	Slot Clock Configuration. Loaded to the PCIe Link Status register.
OBFF_SUP	16:15	0x0	RW	Loaded to the <i>OBFF Supported</i> field in the PCIe Device Capabilities 2 register. Must be set to 0b in the NVM (OBFF is not supported).
Reserved	31:17	0x0	Reserved	Reserved.

10.2.1.2.24 PCIe Power Data Register - GLPCI_PWRDATA (0x000BE490)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
D0_POWER	7:0	0x0	RW	The value in this field is reflected in the PCI Power Management Data register of the LAN functions for D0 power consumption and dissipation (<i>Data_Select</i> = 0 or 4).
COMM_POWER	15:8	0x0	RW	The value in this field is reflected in the PCI Power Management Data register of function 0 when the <i>Data_Select</i> field is set to 8 (common function).
D3_POWER	23:16	0x0	RW	The value in this field is reflected in the PCI Power Management Data register of the LAN functions for D3 power consumption and dissipation (<i>Data_Select</i> = 3 or 7).
DATA_SCALE	25:24	0x0	RW	The value in this field reflects the <i>Data_Scale</i> field in the PCI PMCSR register.
Reserved	31:26	0x0	Reserved	Reserved.

10.2.1.2.25 PCIe PF Device ID - PFPCI_DEVID[PF] (0x000BE080 + 0x4*PF, PF=0...15)

Contains the per-PF Device ID.

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PF_DEV_ID	15:0	0x154B	RW	Contains the device ID for this PF.
VF_DEV_ID	31:16	0x154C	RW	Contains the device ID for the VFs of this PF.



10.2.1.2.26 PCIe Revision ID - GLPCI_REVID (0x000BE4B4)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
NVM_REVID	7:0	0x00	RW	Value of the Rev ID loaded from the NVM.
Reserved	31:8	0x00	Reserved	Reserved.

10.2.1.2.27 PCIe Subsystem ID - GLPCI_SUBVENID (0x000BE48C)

Reset Source: PCIR

NVM Load: Type1/2 in PCIR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
SUB_VEN_ID	15:0	0x8086	RW	Loaded to the PCI configuration Subsystem Vendor ID register.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.2.28 PCIe Statistic Control Register #1 - GLPCI_GSCL_1 (0x0009C48C)

This register controls the operation of the PCIe performance counters.

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GIO_COUNT_EN_0	0	0b	RW	Enables PCIe statistic counter number 0.
GIO_COUNT_EN_1	1	0b	RW	Enables PCIe statistic counter number 1.
GIO_COUNT_EN_2	2	0b	RW	Enables PCIe statistic counter number 2.
GIO_COUNT_EN_3	3	0b	RW	Enables PCIe statistic counter number 3.
LBC_ENABLE_0	4	0b	RW	When set, statistics counter 0 operates in leaky bucket mode. When cleared, leaky bucket mode is disabled and the counter is incremented by one for each event.
LBC_ENABLE_1	5	0b	RW	When set, statistics counter 1 operates in leaky bucket mode. When cleared, leaky bucket mode is disabled and the counter is incremented by one for each event.
LBC_ENABLE_2	6	0b	RW	When set, statistics counter 2 operates in leaky bucket mode. When cleared, leaky bucket mode is disabled and the counter is incremented by one for each event.
LBC_ENABLE_3	7	0b	RW	When set, statistics counter 3 operates in leaky bucket mode. When cleared, leaky bucket mode is disabled and the counter is incremented by one for each event.
Reserved	13:8	0x0	Reserved	Reserved.
Reserved	27:20	0x0	Reserved	Reserved.
GIO_64_BIT_EN	28	0b	RW	Enables two 64-bit counters instead of four 32-bit counters.
GIO_COUNT_RESET	29	0b	RW1S	Reset indication of PCIe statistic counters. Reading this bit returns a 0b.
GIO_COUNT_STOP	30	0b	RW1S	Stop indication of PCIe statistic counters. Reading this bit returns a 0b.
GIO_COUNT_START	31	0b	RW1S	Start indication of PCIe statistic counters. Reading this bit returns a 0b.



10.2.1.2.29 PCIe* Statistic Control Registers #2 - GLPCI_GSCL_2 (0x0009C490)

This register defines the events counted by the performance counters.

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GIO_EVENT_NUM_0	7:0	0x0	RW	Event number that counter 0 counts (GSCN_0).
GIO_EVENT_NUM_1	15:8	0x0	RW	Event number that counter 1 counts (GSCN_1).
GIO_EVENT_NUM_2	23:16	0x0	RW	Event number that counter 2 counts (GSCN_2).
GIO_EVENT_NUM_3	31:24	0x0	RW	Event number that counter 3 counts (GSCN_3).

10.2.1.2.30 PCIe* Statistic Control Register #5...#8 - GLPCI_GSCL_5_8[n] (0x0009C494 + 0x4*n, n=0...3)

These registers control the operation of the leaky bucket counter n.

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LBC_THRESHOLD_N	15:0	0x00	RW	Threshold for the leaky bucket counter n.
LBC_TIMER_N	31:16	0x00	RW	Time period between decrementing the value in leaky bucket Counter n. The time period is defined in us units.

10.2.1.2.31 PCIe* Statistic Counter Registers #0...#3 - GLPCI_GSCN_0_3[n] (0x0009C4A4 + 0x4*n, n=0...3)

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
EVENT_COUNTER	31:0	0x0	RO	A 32-bit event counter. See the section on Performance and Statistics Counters. These registers are stuck at their maximum value of 0xFF...F.



10.2.1.2.32 PCIe Byte Counter Low - GLPCI_BYTCTL (0x0009C488)

A byte counter used by the PCIe performance counters.

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PCI_COUNT_BW_BCT	31:0	0x00	RO	This register contains the low double word of a 64-bit counter that counts PCIe payload bytes. This register gets stuck at its maximum value of 0xFF...F.

10.2.1.2.33 PCIe Byte Counter High - GLPCI_BYTCTH (0x0009C484)

A byte counter used by the PCIe performance counters.

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PCI_COUNT_BW_BCT	31:0	0x00	RO	This register contains the high double word of a 64-bit counter that counts PCIe payload bytes. This register gets stuck at its maximum value of 0xFF...F.

10.2.1.2.34 PCIe Regs Spare Bits 1 - GLPCI_SPARE_BITS_1 (0x0009C4FC)

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
SPARE_BITS	31:0	0x0	RW	Spare Bits.

10.2.1.2.35 PCIe Regs Spare Bits 0 - GLPCI_SPARE_BITS_0 (0x0009C4F8)

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
SPARE_BITS	31:0	0x0	RW	Spare Bits.



10.2.1.2.36 PCIe Packet Counter - GLPCI_PKTCT (0x0009C4BC)

A packet counter used by the PCIe performance counters.

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PCI_COUNT_BW_PCT	31:0	0x0	RO	A 32-bit counter that counts PCIe packets. This register gets stuck at its maximum value of 0xFF...F.

10.2.1.2.37 PCIe PQs Max Used Space - GLPCI_PQ_MAX_USED_SPC (0x0009C4EC)

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GLPCI_PQ_MAX_USED_SPC_12	7:0	0x0	RO	
GLPCI_PQ_MAX_USED_SPC_13	15:8	0x0	RO	
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.2.38 PCIe Mux Selector For PFB - GLPCI_PM_MUX_PFB (0x0009C4F0)

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PFB_PORT_SEL	4:0	0x0	RW	
RSVD_0	15:5	0x0	Reserved	Reserved.
INNER_PORT_SEL	18:16	0x0	RW	
RSVD_1	31:19	0x0	Reserved	Reserved.



10.2.1.2.39 PCIe Mux Selector For NPQs - GLPCI_PM_MUX_NPQ (0x0009C4F4)

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
NPQ_NUM_PORT_SEL	2:0	0x0	RW	
RSVD_0	15:3	0x0	Reserved	Reserved.
INNER_NPQ_SEL	20:16	0x0	RW	
RSVD_1	31:21	0x0	Reserved	Reserved.

10.2.1.2.40 PCIe Upper Address - GLPCI_UPADD (0x000BE4F8)

This register is used to block PCIe master accesses above some address. See the section Blocking on Upper Address.

Reset Source: PCIR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
Reserved	0	0b	Reserved	Reserved.
ADDRESS	31:1	0x0	RW	Address. Bits [31:1] correspond to bits [63:33] in the PCIe address space, respectively.

10.2.1.2.41 PCIe LCB Address Port - GLPCI_LCBADD (0x0009C4C0)

Note that this register is accessible only in debug mode.

Reset Source: PCIR

NVM Load: Forbidden Load

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ADDRESS	17:0	0x0	RW	An 18-bit address register, part of a port, used to load NVM configuration into the sub-units of the PCIe unit. Operates together with the GLPCI_LCBDATA register.
Reserved	19:18	0x0	Reserved	Reserved.



Field	Bit(s)	Init.	Access Type	Description
BLOCK_ID	30:20	0x0	RW	An ID of a sub-unit in the PCIe unit. Supported values are: // 0. NPQ: RLAN. // 1. NPQ: TLAN. // 2. NPQ: RX_PE. // 3. NPQ: TX_PE. // 4. NPQ: PMAT. // 5. NPQ: MNG. // 6. NPQ: TDPU. // 7. PQ: RLAN. // 8. PQ: TLAN. // 9. PQ: RX_PE. // 10. PQ: TX_PE. // 11. PQ: PMAT. // 12. PQ: MNG. // 13. PQ: RDPU. // 14. VDM. // 15. Don't care. // 16. HIU: CFG/Slave/MSG/MSI-X. //126: LCB's internal config space registers. //127: LCB's internal memory space registers.
LOCK	31	0b	RO	This bit serves as a semaphore between multiple agents accessing the LCB port. Hardware functionality: 1. Reading the bit sets it to 1b (if already at 1b, then no effect). 2. Writing to the bit has no effect (it is RO). 3. Accessing the GLPCI_LCBDATA register clears this bit to 0b. Recommended flow: 1. Read the <i>Lock</i> bit. 2. If returned value is 0b, proceed. Else, try again later. 3. Write an LCB address to the GLPCI_LCBADDR register. 4. Write to or read from the GLPCI_LCBDATA register. Repeat this flow for each 32-bit access to the LCB.

10.2.1.2.42 PCIe LCB Data Port - GLPCI_LCBDATA (0x0009C4C4)

Note that this register is accessible only in debug mode.

Reset Source: PCIR

NVM Load: Forbidden Load

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LCB_DATA	31:0	0x0	RW	A 32-bit data register, part of a port, used to load the NVM configuration into the PCIe LCB unit. Operates together with the GLPCI_LCBADDR register.

**10.2.1.2.43 PF PCIe Configuration Indirect Access Address - PF_PCI_CIAA[PF] (0x0009C080 + 0x4*PF, PF=0...15)**

Reset Source: FLR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ADDRESS	11:0	0x0	RW	The configuration space address to access.
VF_NUM	18:12	0x0	RW	Defines the VF number to access. The VF number is the absolute VF number in the device. Note that the PF can access any VF (its own VFs as well as other VFs).
Reserved	31:19	0x0	Reserved	Reserved. Ignore on read; write 0b.

10.2.1.2.44 PCIe Configuration Indirect Access Data - PF_PCI_CIAD[PF] (0x0009C100 + 0x4*PF, PF=0...15)

Reset Source: FLR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
DATA	31:0	0x0	RW	This register is used to access the configuration registers of the VF. It operates together with the PF_PCI_CIAA register as follows: Reading this register returns the content of the register at offset = ADDRESS in the configuration space of VF index = VF_NUM (the ADDRESS and VF_NUM parameters are defined by the PF_PCI_CIAA register). Writing to this register is gated by the <i>Config_access_enable</i> flag in the GL_PCI_DBGCTL register. If enabled, the value written to this register is programmed to the register at offset = ADDRESS in the configuration space of VF index = VF_NUM.

10.2.1.2.45 PCIe VM Pending Index - PFPCI_VMINDEX[PF] (0x0009C300 + 0x4*PF, PF=0...15)

Reset Source: PCIR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
VMINDEX	8:0	0x0	RW	VM Index. Software sets <i>VMINDEX</i> that its transaction pending flag should be reflected in the PFPCI_VMPEND register. The VM index is an absolute index in the range of 0 through 383. It can only be set by software to VMs that the PF owns and only to VMs that are not assigned to a VF.
Reserved	31:9	0x0	Reserved	Reserved.



10.2.1.2.46 PCIe VF Flush Done - PFPCI_VF_FLUSH_DONE[VF] (0x0009C600 + 0x4*VF, VF=0...127)

Reset Source: PCIR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
FLUSH_DONE	0	0b	RO	Flush Done.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.2.47 PCIe PF Flush Done - PFPCI_PF_FLUSH_DONE[PF] (0x0009C800 + 0x4*PF, PF=0...15)

Reset Source: PCIR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FLUSH_DONE	0	0b	RO	Flush Done.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.2.48 PCIe VM Flush Done - PFPCI_VM_FLUSH_DONE[PF] (0x0009C880 + 0x4*PF, PF=0...15)

Reset Source: PCIR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FLUSH_DONE	0	0b	RO	Flush Done.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.2.49 PCIe VM Pending Status - PFPCI_VMPEND[PF] (0x0009C380 + 0x4*PF, PF=0...15)

Reset Source: PCIR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PENDING	0	0b	RO	PCIe Transaction Pending Status. The reported VM is controlled by the <i>VMINDEX</i> field in the PFPCI_VMINDEX register. This flag is set to 1b as long as there is at least one PCIe transaction pending for its completion.
Reserved	31:1	0x0	Reserved	Reserved.



10.2.1.3 MAC Registers

10.2.1.3.1 HSEC CONTROL Receive PAUSE_DA_UCAST_PART2 - PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART2[PRT2] (0x001E3120 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_PAUSE_DA_UCAST_PART2	15:0	0x0	RW	Unicast Destination Address For Pause Processing. Valid only if the UC address is enabled for control processing through RX_CHECK_UC_PPP/PCP/GPP/GCP.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.3.2 HSEC CONTROL Receive PFC ENABLE - PRTMAC_HSEC_CTL_RX_PAUSE_ENABLE[PRT2] (0x001E30C0 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_PAUSE_ENABLE	8:0	0x0	RW	Rx Priority Flow Control Enable. This field is used to enable priority flow control per priority. When bit[x] is set to 1b, PFC processing is enabled for priority-x. Bit[8] Is used to enable 802.3x flow control. Note: Priority flow control can be enabled only when the receive packet buffer of the port is configured for DCB mode.
Reserved	31:9	0x0	Reserved	Reserved.

10.2.1.3.3 HSEC CONTROL Receive ENABLE_GPP - PRTMAC_HSEC_CTL_RX_ENABLE_GPP[PRT2] (0x001E3260 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_ENABLE_GPP	0	0b	RW	A value of 1b enables 802.3x pause packet processing. Note: 802.3x pause is also referred to as global pause in HSEC registers.
Reserved	31:1	0x0	Reserved	Reserved.



10.2.1.3.4 HSEC CONTROL Transmit PAUSE_ENABLE - PRTMAC_HSEC_CTL_TX_PAUSE_ENABLE[PRT2] (0x001E30D0 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_TX_PAUSE_ENABLE	8:0	0x0	RW	Tx Priority Flow Control Enable. This field is used to enable priority flow control per priority. When bit[x] is set to 1b, PFC packet transmission is enabled for priority-X. Bit[8] is used to enable 802.3x flow control.
Reserved	31:9	0x0	Reserved	Reserved.

10.2.1.3.5 HSEC CONTROL Receive PAUSE_DA_UCAST_PART1 - PRTMAC_HSEC_CTL_RX_PAUSE_DA_UCAST_PART1[PRT2] (0x001E3110 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_PAUSE_DA_UCAST_PART1	31:0	0x0	RW	Unicast Destination Address For Pause Processing. Valid only if the UC address is enabled for control processing through RX_CHECK_UC_PPP/PCP/GPP/GCP.

10.2.1.3.6 HSEC CONTROL Receive FORWARD_CONTROL - PRTMAC_HSEC_CTL_RX_FORWARD_CONTROL[PRT2] (0x001E3360 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_FORWARD_CONTROL	0	0b	RW	A value of 1b indicates that HSEC MAC forwards control packets to the user. A value of 0b causes the HSEC MAC to drop control packets.
Reserved	31:1	0x0	Reserved	Reserved.



10.2.1.3.7 HSEC CONTROL Transmit SA_GPP_PART1 - PRTMAC_HSEC_CTL_TX_SA_PART1[PRT2] (0x001E34B0 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_TX_SA_PART1	31:0	0x0	RW	Source address for transmitting pause packets.

10.2.1.3.8 HSEC CONTROL Receive PAUSE_SA_PART2 - PRTMAC_HSEC_CTL_RX_PAUSE_SA_PART2[PRT2] (0x001E3150 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: No

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_PAUSE_SA_PART2	15:0	0x0	RW	Source address for pause processing.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.3.9 HSEC CONTROL Receive PAUSE_SA_PART1 - PRTMAC_HSEC_CTL_RX_PAUSE_SA_PART1[PRT2] (0x001E3140 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: No

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_PAUSE_SA_PART1	31:0	0x0	RW	Source address for pause processing.

10.2.1.3.10 HSEC CONTROL Transmit SA_GPP_PART2 - PRTMAC_HSEC_CTL_TX_SA_PART2[PRT2] (0x001E34C0 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2



Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_TX_SA_PART2	15:0	0x0	RW	Source address for transmitting pause packets.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.3.11 HSEC CONTROL Receive ENABLE_GCP - PRTMAC_HSEC_CTL_RX_ENABLE_GCP[PRT2] (0x001E30E0 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_ENABLE_GCP	0	1b	RW	When set to 1b and rx_forward_control is set to 0b, flow control packets are terminated by HSEC MAC. When set to 0b, both 802.3x and PFC packet processing is disabled and HSEC MAC forwards all control packets. Note: In order to enable 802.3x and PFC functional packet processing, a dedicated enable bit must be configured.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.3.12 HSEC CONTROL Transmit PAUSE_QUANTA - PRTMAC_HSEC_CTL_TX_PAUSE_QUANTA[n,PRT2] (0x001E3370 + 0x10*n + 0x4*PRT2, n=0...8, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_TX_PAUSE_QUANTA	15:0	0xFFFF	RW	These nine buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for stat_tx_pause_quanta[8] is used for global pause operation. All other values are used for priority pause operation.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.3.13 HSEC CONTROL Transmit PAUSE_REFRESH_TIMER - PRTMAC_HSEC_CTL_TX_PAUSE_REFRESH_TIMER[n,PRT2] (0x001E3400 + 0x10*n + 0x4*PRT2, n=0...8, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_TX_PAUSE_REFRESH_TIMER	15:0	0x0	RW	These nine buses set the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for stat_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.3.14 HSEC CONTROL Receive ENABLE_PPP - PRTMAC_HSEC_CTL_RX_ENABLE_PPP[PRT2] (0x001E32E0 + 0x4*PRT2, PRT2=0...1)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT2

Field	Bit(s)	Init.	Access Type	Description
HSEC_CTL_RX_ENABLE_PPP	0	0b	RW	A value of 1b enables priority pause packet processing.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.3.15 Port MAC Address High - PRTGL_SAH[PRT] (0x001E2140 + 0x4*PRT, PRT=0...3)

This register contains the upper bits of the NVM pre-loaded 48-bit port Ethernet MAC address. The complete address is {PRTGL_SAH, PRTGL_SAL} and also contains the port's maximum frame size configuration.

The register contents is loaded from the NVM and its location in the NVM is pointed by the 7th item in the Auto-Generated Pointers module. It is a per-device, per-enabled port manufacturing value.

In SFP NVM images, it can be set identical to the concatenated [PFPM_SAH1|PFPM_SAH0] words of the EMP Settings module of PFs 0...3.

In MFP NVM images, it should be separate addresses than the PFPM_SAH addresses.

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT



Field	Bit(s)	Init.	Access Type	Description
FC_SAH	15:0	0x0	RW	Station address high used for flow control packet processing. The upper 16 bits of the 48-bit Ethernet MAC address. Note: This field is defined in big Endian (MS byte of SAH is last on the wire).
MFS	31:16	0x2600	RW	This field defines the maximum receive frame size in bytes from MAC addresses up to and inclusive of CRC. Frames received that are larger than this value might be dropped based on the SBP configuration. Note: This configuration has no effect on maximum transmit frame size.

10.2.1.3.16 Port MAC Address Low - PRTGL_SAL[PRT] (0x001E2120 + 0x4*PRT, PRT=0...3)

This register contains the NVM loaded lower bits of the 48-bit port Ethernet MAC Address. Its contents is loaded from the NVM and its location in the NVM is pointed by the 6th item in the Auto-Generated Pointers module. It is a per-device, per-enabled port manufacturing value.

In SFP NVM images, it can be set identical to the concatenated [PFPM_SAL1|PFPM_SAL0] words of the EMP Settings module of PFs 0...3. In MFP NVM images, it should be separate addresses than the PFPM_SAL addresses.

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
FC_SAL	31:0	0x0	RW	Station address low used for flow control packet processing. The lower 32 bits of the 48-bit Ethernet MAC address. Note: This field is defined in big Endian (LS byte of SAL is first on the wire).

10.2.1.3.17 PCS_XAUI_SWAP_B - PRTMAC_PCS_XAUI_SWAP_B (0x0008C484)

Enables lane swapping, when operating in XAUI mode.

This register performs mapping between Group-B, on board, lanes 0-3 to internal PCS/MAC lanes.

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
SWAP_TX_LANE3	1:0	0x3	RW	Selects the physical lane number to output MAC Lane3. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.



Field	Bit(s)	Init.	Access Type	Description
SWAP_TX_LANE2	3:2	0x2	RW	Selects the physical lane number to output MAC Lane2. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_TX_LANE1	5:4	0x1	RW	Selects the physical lane number to output MAC Lane1. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_TX_LANE0	7:6	0x0	RW	Selects the physical lane number to output MAC Lane0. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_RX_LANE3	9:8	0x3	RW	Selects the physical lane number to receive data from into MAC Lane3. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_RX_LANE2	11:10	0x2	RW	Selects the physical lane number to receive data from into MAC Lane2. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_RX_LANE1	13:12	0x1	RW	Selects the physical lane number to receive data from into MAC Lane1. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_RX_LANE0	15:14	0x0	RW	Selects the physical lane number to receive data from into MAC Lane0. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.3.18 PCS_XAUI_SWAP_A - PRTMAC_PCS_XAUI_SWAP_A (0x0008C480)

Enables lane swapping, when operating in XAUI mode.

This register performs mapping between Group-A, on board, lanes 0-3 to internal PCS/MAC lanes.

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
SWAP_TX_LANE3	1:0	0x3	RW	Selects the physical lane number to output MAC Lane3. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_TX_LANE2	3:2	0x2	RW	Selects the physical lane number to output MAC Lane2. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_TX_LANE1	5:4	0x1	RW	Selects the physical lane number to output MAC Lane1. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_TX_LANE0	7:6	0x0	RW	Selects the physical lane number to output MAC Lane0. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_RX_LANE3	9:8	0x3	RW	Selects the physical lane number to receive data from into MAC Lane3. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_RX_LANE2	11:10	0x2	RW	Selects the physical lane number to receive data from into MAC Lane2. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_RX_LANE1	13:12	0x1	RW	Selects the physical lane number to receive data from into MAC Lane1. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
SWAP_RX_LANE0	15:14	0x0	RW	Selects the physical lane number to receive data from into MAC Lane0. 0x0 = Lane0. 0x1 = Lane1. 0x2 = Lane2. 0x3 = Lane3.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.4 Power Management Registers

Registers related to DMA coalescing and EEE.

10.2.1.4.1 General Control - PRTPM_GC[PRT] (0x000B8140 + 0x4*PRT, PRT=0...3)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
EMP_LINK_ON	0	0b	RW	EMP Link On. If set to 0b, this Ethernet port is not required for EMP functionality. If set to 1b, this Ethernet port is required to be up for EMP functionality.
MNG_VETO	1	0b	RW	MNG_VETO manageability veto for link LPLU and reset (access read/write by manageability and read only to host). Impact on LPLU: 0b = No specific constraints on link from manageability. 1b = Hold off any low-power link mode changes. This is done to avoid link loss and interrupting manageability activity. Impact on reset: Reset impact is global for the device and is received as a logical OR between the per port bits. When at least one of the per port bits is set, PCI reset triggers internal CORER, which does not impact the MAC and PHY interfaces. When all bits are cleared, PCI reset triggers internal GLOBR, which also initialize the MAC and PHY interfaces.
RATD	2	0b	RW	Restarts Auto-negotiation on Transition to Dx. This bit enables the functionality to restart KX/KR backplane auto-negotiation on transitions to Dx (Dr/D3). 0b = Does not restart auto-negotiation when all port's functions moves to the Dx state. 1b = Restarts auto-negotiation to reach a low-power link mode (1 GbE link) when all port's functions transitions to the Dx state.
LCDMP	3	0b	RW	Lowest Common Denominator (LCD) on Dx (Dr/D3) without main power. 0b = No specific action. 1b = Move to lowest common denominator link speed when main power is removed. When the RATD bit is also set to 1b, it causes the link mode to auto-negotiate to the lowest common denominator (if enabled) when main power (MAIN_PWR_OK) is removed.
Reserved	30:4	0x0	Reserved	Reserved.
LPLU_ASSERTED	31	0b	RW	LPLU asserted. This bit is set/cleared by firmware according to the LPLU enable/disable state set by the Set PHY Config AQC received from the PF(s) attached to the port.



10.2.1.4.2 Energy Efficient Ethernet (EEE) Register - PRTPM_EEER[PRT] (0x001E4360 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TW_SYSTEM	15:0	0x0	RW	<p>Time expressed in microseconds that no data is transmitted following a move from an EEE Tx LPI link state to link active state. The field holds the transmit Tw_sys_tx value negotiated during EEE LLDP negotiation.</p> <p>Notes:</p> <ol style="list-style-type: none"> If the value is lower than minimum a Tw_sys_tx value defined in IEEE Std. 802.3az clause 78.5 then interval where no data is transmitted following a move out of an EEE Tx LPI state defaults to a minimum Tw_sys_tx. Following link disconnect or auto-negotiation, the value of this field returns to the default value until software re-negotiates a new tw_sys_tx value via EEE LLDP. A fast retrain and local/remote fault indication are not considered a link disconnect and do not cause the field to return to the default value. When transmitting flow control frames, the device waits the minimum time defined in the IEEE Std. 802.3az standard before transmitting the flow control packet. The device does not wait the Tw_system time following an exit of LPI before transmitting the flow control frame.
TX_LPI_EN	16	0b	RW	<p>Enable entry into EEE LPI on the Tx path. 0b = Disable entry into EEE LPI on the Tx path. 1b = Enable entry into EEE LPI on Tx path.</p> <p>Notes:</p> <ol style="list-style-type: none"> Even when TX_LPI_EN is 1b, the device does not enable entry into a Tx LPI state for at least a PRTPM_EEEC.TX_LU_LPI_DLY following the change of link_status to OK as defined in IEEE Std. 802.3az clause 78.1.2.1. Even if the TX_LPI_EN bit is set, the device initiates entry into a Tx EEE LPI link state only if EEE support at the link speed was negotiated during auto-negotiation or forced by software to enable EEE on non-auto-negotiation protocols.
Reserved	31:17	0x0	Reserved	<p>Reserved. Write 0, ignore on read.</p>



10.2.1.4.3 Energy Efficient Ethernet (EEE) Control - PRTPM_EEE[PART] (0x001E4380 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	15:0	0x0	Reserved	Reserved. Write 0, ignore on read.
TW_WAKE_MIN	21:16	0xA	RW	Minimum time expressed in 1 μ s between sending a request to move into EEE Tx LPI and sending a request to move back to an active state. Note: If conditions to exit LPI during the Tw_wake_min interval cease to exist, then the device does not move out of Tx LPI after the timer has expired.
Reserved	23:22	0x0	Reserved	Reserved.
TX_LU_LPI_DLY	25:24	0x3	RW	Delay to enable entering a Tx EEE LPI state following a link-up indication. 00b = No delay. 01b = 10 μ s. 10b = 100 μ s. 11b = 1 second. Note: IEEE Std. 802.3az clause 78.1.2.1 defines a delay of 1 second following link-up.
TEEE_DLY	31:26	0x2	RW	TX EEE LPI Entry delay Field defines delay to EEE entry once conditions to enter EEE LPI are detected. Field resolution is 1's. Notes: 1. If conditions to enter LPI during the TEEE_DLY interval cease to exist, the device does not enter Tx LPI and continues normal operation. 2. Minimum configuration should be 0x1.

10.2.1.4.4 Energy Efficient Ethernet (EEE) STATUS - PRTPM_EEE_STAT[PART] (0x001E4320 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	28:0	0x0	Reserved	Reserved. Write 0, ignore on read.
EEE_NEG	29	0b	RO	EEE Support Negotiated on Link. 0b = EEE operation not supported on link. 1b = EEE operation supported on link.
RX_LPI_STATUS	30	0b	RO	Rx Link in LPI state. 0b = Rx in active state. 1b = Rx in LPI state.



Field	Bit(s)	Init.	Access Type	Description
TX_LPI_STATUS	31	0b	RO	Tx Link in LPI State. 0b = Tx in active state. 1b = Tx in LPI state.

10.2.1.4.5 EEE Tx Firmware Done - PRTPM_EEEFWD[PRT] (0x001E4400 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	30:0	0x0	Reserved	Reserved.
EEE_FW_CONFIG_DONE	31	0b	RW	Set by firmware to indicate a firmware configuration of the EEE parameters after link establishment completes. Cleared by hardware when the link is down.

10.2.1.4.6 EEE Tx Control - PRTPM_EEETXC[PRT] (0x001E43E0 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TW_PHY	15:0	0x1	RW	The Tw_phy value is set by firmware and should accommodate for the time defined in IEEE Std. 802.3az for the per connected PHY technology Tw_phy with the addition of the proper per-PHY technology addition as defined in the GLPM_EEE_SU and GLPM_EEE_SU_EXT registers. Value defined in this field is expressed in 102.4 ns resolution. Note: The idle time value defined by this field is used when moving out of a EEE Tx LPI state to transmit flow control frames even if the value specified in the EEER.Tw_system field is higher.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.4.7 EEE Tx LPI Count - PRTPM_TLPIC[PRT] (0x001E43C0 + 0x4*PRT, PRT=0...3)

This register counts EEE TX LPI entry events. A EEE TX LPI event occurs when the transmitter enters a EEE (IEEE Std. 802.3az) LPI state. This register only increments if transmits are enabled and EEE operation is enabled.

This register is cleared on read.

Reset Source: GLOBR

NVM Load: No

Scope: PRT



Field	Bit(s)	Init.	Access Type	Description
ETLPIC	31:0	0x0	RO	Number of EEE Tx LPI events.

10.2.1.4.8 EEE Rx LPI Count - PRTPM_RLPIC[PRT] (0x001E43A0 + 0x4*PRT, PRT=0...3)

This register counts EEE RX LPI entry events. A EEE RX LPI event occurs when the receiver detects a link partner entry into a EEE (IEEE Std. 802.3az) LPI state. This register only increments if receives are enabled and EEE operation is enabled.

This register is cleared on read.

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
ERLPIC	31:0	0x0	RO	Number of EEE Rx LPI events.

10.2.1.5 Wake-Up and Proxying Registers

10.2.1.5.1 WU on MNG Control - GLPM_WUMC (0x0006C800)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
NOTCO	0	0b	RW	Ignore management only packets for wake up. 0b = Ignore management only packets for wake up. A packet that meets the criteria defined in the MNGONLY register (such as only intended for the MC and not the host) does not wake the host. 1b = Wake the host on any WUFC matched packet. Note: This bit impacts the wake up decision only when all PF functions are in D3 power state.
Reserved	15:3	0x0	RW	Reserved.
MNG_WU_PF	31:16	0x0	RW	These bits can set a bit to indicate a manageability-initiated wake-up event (bit per PF).

10.2.1.5.2 MAC Address Low - PRTPM_SAL[n,PRT] (0x001E4440 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

This register contains the station address lower 32 bits of the 48-bit Ethernet MAC address. It is loaded by firmware from the NVM and can be altered later by a PF driver admin command.

Reset Source: PFR

NVM Load: Firmware loaded

Scope: PRT



Field	Bit(s)	Init.	Access Type	Description
PFPM_SAL	31:0	0x0	RW	Station Address Low. The lower 32 bits of the 48-bit NVM pre-assigned Ethernet MAC address. Note: This field is defined in big Endian (LS byte of SAL is first on the wire).

10.2.1.5.3 MAC Address High - PRTPM_SAH[n,PRT] (0x001E44C0 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

This register contains the station address upper 16 bits of the 48-bit Ethernet MAC address. It is loaded by firmware from the NVM and can be altered later by a PF driver admin command.

PRTPM_SAH.AV determines whether this address is valid and compared against the incoming packet. After reset, firmware loads the relevant MAC address from the NVM, and sets its *Address Valid* field to 1b.

Reset Source: PFR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
PFPM_SAH	15:0	0x0	RW	Station Address High. The upper 16 bits of the 48-bit Ethernet MAC address. Note: This field is defined in big Endian (MS byte of PRTPM_SAH is Last on the wire).
Reserved	25:16	0x0	Reserved	Reserved.
PF_NUM	29:26	0x0	RW	PF number used for reporting a PF wake up. This value is written by firmware.
MC_MAG_EN	30	0b	RW	Enable promiscuous multicast for magic packets. If this bit is set to 1b, every multicast magic packet generates a WoL event if enabled in PFPM_WUFC.MAG or in PFPM_APM.APME.
AV	31	0b	RW	Address Valid. If the NVM is present, the station address is assigned by firmware after loading from the NVM and its <i>Address Valid</i> field is set to 1b.

10.2.1.5.4 Flexible Host Filter Header Removal - PRTPM_FHFHR[PRT] (0x0006C000 + 0x4*PRT, PRT=0...3)

Note: This register is not supported.

The Flexible Host Filter Header Removal (FHFHR) register defines the packet headers that need to be removed before doing a comparison via the Flex Filter Tables (FHFT registers).

Notes:

1. This register is reset by LAN_PWR_GOOD only.
2. Register is per port.
3. Register should be read by the PF driver prior to FHFT programming.

Reset Source: POR

NVM Load: No

Scope: PRT



Field	Bit(s)	Init.	Access Type	Description
UNICAST	0	1b	RW	0b = Compare unicast-TAG information in the flex filters. 1b = Remove the unicast-TAG field before doing a comparison in the flex filters.
MULTICAST	1	1b	RW	0b = Compare multicast-TAG information in the flex filters. 1b = Remove the multicast-TAG field before doing a comparison in the flex filters.
Reserved	31:2	0x0	Reserved	Reserved.

10.2.1.5.5 APM Control Register - PFPM_APM[PF] (0x000B8080 + 0x4*PF, PF=0...15)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
APME	0	0b	RW	Advance Power Management Enable. If set to 1b, APM wakeup is enabled. Note: This bit is reset on power-on reset (LAN_PWR_GOOD) only.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.5.6 Wake Up Control Register - PFPM_WUC[PF] (0x0006B200 + 0x4*PF, PF=0...15)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
Reserved	4:0	0x0	Reserved	Reserved
EN_APM_D0	5	0b	RW	Enable APM wake on D0. 0b = Enable wake only when the function is in D3/Dr. 1b = Enable wake in D0.
Reserved	31:6	0x0	Reserved	Reserved.

10.2.1.5.7 Wake Up Filter Control Register - PFPM_WUFC[PF] (0x0006B400 + 0x4*PF, PF=0...15)

This register is used to enable each of the pre-defined and flexible filters for wake up support. A value of 1b means the filter is turned on; a value of 0b means the filter is turned off.

The bits only reset at power-on reset (LAN_PWR_GOOD) and at D3/Dr to D0 transitions.

Note: If the *NoTCO* bit is set, then any packet that passes manageability packet filtering is defined as *MNG_ONLY* does not cause a wake-up event.

Reset Source: POR

NVM Load: No

Scope: PF



Field	Bit(s)	Init.	Access Type	Description
LNKC	0	0b	RW	Link Status Change Wake-up Enable.
MAG	1	0b	RW	Magic Packet Wake-up Enable.
Reserved	2	0b	Reserved	Reserved.
MNG	3	0b	RW	MNG Wake-up Enable.
Reserved	15:12	0x0	Reserved	Reserved.
Reserved	30:24	0x0	Reserved	Reserved.
FW_RST_WK	31	0b	RW	Enable Wake on Firmware Reset Assertion. When set, a firmware reset causes system wake so that the software device driver can re-send proxying information to firmware.

10.2.1.5.8 Wake Up Status Register - PFPM_WUS[PF] (0x0006B600 + 0x4*PF, PF=0...15)

This register is used to record statistics about all wake-up packets received. If a packet matches multiple criteria then multiple bits could be set. Writing a 1b to any bit clears that bit.

This register is not cleared when PE_RST_N is asserted (except for the PME_STATUS bit). It is only cleared when LAN_PWR_GOOD is de-asserted or when cleared by the software device driver.

Note: If additional packets are received that match one of the wake-up filters, after the original wake-up packet is received, the WUS register is not updated with the new match detection until the register is cleared.

Wakeup statuses are also reported in this register when no wake-up event is generated because a CRC error is detected on the presumed wakeup packet.

Reset Source: POR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
LNKC	0	0b	RW1C	Link Status Changed.
MAG	1	0b	RW1C	Magic Packet Received.
PME_STATUS	2	0b	RW1C	PME_Status. This bit is set when the device receives a wake-up event. It is the same as the PME_Status bit in the PMCSR. Writing a 1b to this bit also clears the PME_Status bit in the PMCSR. This bit is reset only on power-on reset (LAN_PWR_GOOD). When AUX_PWR = 0b, this bit is also reset on de-assertion of PE_RST_N.
MNG	3	0b	RW1C	MNG Wake-up Status.
Reserved	15:4	0x0	Reserved	Reserved.
Reserved	30:24	0x0	Reserved	Reserved.
FW_RST_WK	31	0b	RW1C	Wake due to firmware reset assertion event. When set to 1b, indicates that firmware reset assertion caused system wake so the software device driver can re-send proxying information to firmware.



10.2.1.6 NVM Registers

10.2.1.6.1 Flash ID Register - GLNVM_FLASHID (0x000B6104)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FLASHID	23:0	0x0	RO	Flash ID. It is formed by the 3-bytes JEDEC-ID of the device. Byte 0 (bits 7:0) is the first byte read from the flash after the READ JEDEC-ID Opcode (0x9F) was issued to it. It contains the Manufacturer's ID. Byte 1 (bits 15:8) is the second byte read from the flash after the READ JEDEC-ID Opcode (0x9F) was issued to it. It contains the Memory Type. Byte 2 (bits 23:16) is the third byte read from the flash after the READ JEDEC-ID Opcode (0x9F) was issued to it. It contains the Memory Capacity.
Reserved	30:24	0x0	Reserved	Reserved.
FLEEP_PERF	31	0b	RW	FLEEP Block Performance Enhancement. This bit enables a 4-byte transaction to the SPI Flash when the host reads from the Flash through the expansion ROM BAR interface. Bit set to 1b = 4-byte transactions are enabled. Bit set to 0b = Only a single byte transactions take place. This bit can be changed any time between transactions.



10.2.1.6.2 Global NVM General Status Register - GLNVM_GENS (0x000B6100)

This register cannot be loaded from the NVM via one of the CSR format modules.

Reset Source: POR

NVM Load: Forbidden No Load

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
NVM_PRES	0	0b	RO	NVM Present. Setting this bit to 1b indicates that a Flash part is present and that a correct validity field was found in one of the two basic banks, such as a validity field value read is 01b.
Reserved	4:1	0x0	Reserved	Reserved.
SR_SIZE	7:5	0x6	RO	Shadow RAM Size. This field defines the size of the internal shadow RAM. This is equal to the size of the internal shadow RAM in power of 2 KB units. Initial value is 110b = 64 KB.
BANK1VAL	8	0b	RW	Basic Bank 1 Valid. When set to 1b, indicates that the content of the basic bank 1 of the Flash device is valid. When set to 0b, indicates that the content of basic banks 0 is valid. Meaningful only when the NVM_PRES bit is read as 1b. It is written by hardware once at power up and then toggled only by the EMP.
Reserved	22:9	0x0	Reserved	Reserved.
Reserved	24	0b	Reserved	Reserved.
Reserved	31:26	0x0	Reserved	Reserved.

10.2.1.6.3 Flash Access Register - GLNVM_FL_A (0x000B6108)

This register can be written by the host only when the device is in blank Flash programming mode. It cannot be loaded from the NVM via one of the CSR format modules.

Reset Source: POR

NVM Load: Forbidden No Load

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FL_SCK	0	0b	RW	Clock input to the Flash. When FL_GNT is set to 1b, the FL_SCK output signal is mapped to this bit and provides the serial clock input to the Flash. Software clocks the Flash via toggling this bit with successive writes. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.
FL_CE	1	0b	RW	Chip select input to the Flash. When FL_GNT is set to 1b, the FL_CE output signal is mapped to the chip select of the Flash device. Software enables the Flash by writing a 0b to this bit. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.



Field	Bit(s)	Init.	Access Type	Description
FL_SI	2	0b	RW	Data input to the Flash. When FL_GNT is set to 1b, the FL_SI output signal is mapped directly to this bit. Software provides data input to the Flash via writes to this bit. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.
FL_SO	3	0b	RW	Data output bit from the Flash. The FL_SO output signal is mapped directly to this bit in the register and contains the Flash serial data output. This bit is read-only from a software perspective. Note that writes to this bit have no effect. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.
FL_REQ	4	0b	RW	Request Flash Access. Software must write a 1b to this bit to get direct Flash access. It has access when FL_GNT is set to 1b. When software completes the access, it must then write a 0b. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.
FL_GNT	5	0b	RO	Grant Flash Access. When this bit is set to 1b, software can access the Flash using the FL_SCK, FL_CE, FL_SI, and FL_SO bits. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.
LOCKED	6	1b	RW	Normal NVM Programming Mode. When set to 1b, the device is in normal NVM programming mode. When set to 0b, the device is in blank Flash programming mode. The bit can be cleared by the EMP or by hardware.
Reserved	17:7	0x0	Reserved	Reserved
FL_SADDR	28:18	0x0	RW	Flash Sector Erase Address. Determines which 4 KB sector is erased when FL_SER command is used. This address is expressed in sector index units, starting from sector index 0. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.
FL_SER	29	0b	Reserved	Flash Sector Erase Command. This bit is auto-cleared and reads as 0b. The 4 KB sector index to be erased is determined by FL_SADDR field. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.
FL_BUSY	30	0b	RO	Flash Busy. This bit is set to 1b while a write or an erase to the Flash is in progress. While this bit is cleared (reads as 0b), software/EMP can access to write a new byte to the Flash. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.
FL_DER	31	0b	RW	Flash Device Erase Command. This bit is auto-cleared and reads as 0b. The entire Flash device is erased. Note: This field is operational to software only when in blank Flash programming mode. It is operational to the EMP in any mode.



10.2.1.6.4 Shadow RAM Control Register - GLNVM_SRCTL (0x000B6110)

This register cannot be loaded from the NVM via one of the CSR format modules.

Reset Source: POR

NVM Load: Forbidden No Load

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
SRBUSY	0	0b	RO	Shadow RAM Busy. This bit indicates that the shadow RAM is busy and could not be accessed.
Reserved	13:1	0x0	Reserved	Reserved.
ADDR	28:14	0x0	RW	Address. This field is written by the host along with the <i>Start</i> bit and the <i>Write</i> bit to indicate which shadow RAM word address to read or write.
WRITE	29	0b	RW	Write. This bit signals the shadow RAM if the current operation is read or write. 0b = Read. 1b = Write.
START	30	0b	RW	Start. Writing a 1b to this bit causes the read or write operation of the shadow RAM according to the write bit. This bit is self cleared by hardware.
DONE	31	1b	RW	Transaction Done. This bit is cleared after the <i>Start</i> bit and <i>Write</i> bit are set by the host and is set back again when the shadow RAM write or read transaction completes.

10.2.1.6.5 Shadow RAM Read/Write Data - GLNVM_SRDATA (0x000B6114)

Reset Source: POR

NVM Load: Forbidden No Load

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
WRDATA	15:0	0x0	RW	Write Data. Data to be written to the shadow RAM.
RDDATA	31:16	0x0	RO	Read Data. Data returned by hardware from the shadow RAM read.



10.2.1.6.6 Unit Load Status - GLNVM_ULD (0x000B6008)

This register provides indications on the completion of loading the shadow RAM and the alternate module into the device units.

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
CONF_PCIR_DONE	0	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_PCIRTL_DONE	1	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_LCB_DONE	2	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_CORE_DONE	3	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_GLOBAL_DONE	4	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_POR_DONE	5	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_PCIE_ANA_DONE	6	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_PHY_ANA_DONE	7	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_EMP_DONE	8	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
CONF_PCIALT_DONE	9	0b	RO	When cleared, indicates that the units affected by the respective NVM module still need to be initialized. When set, indicates that the units affected by the module are ready.
Reserved	31:10	0x0	Reserved	Reserved.



10.2.1.6.7 Protected CSR List - GLNVM_PROTCSR[n] (0x000B6010 + 0x4*n, n=0..59)

Reset Source: POR

NVM Load: Protected

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ADDR_BLOCK	23:0	0xFFFFFFFF	RW	CSR Blocked Address. The field contains the address of a blocked register included in the CSR Protected List NVM module. Blocked registers cannot be loaded from a CSR format module (Type 1/2/3). The register is loaded from shadow RAM at POR events only, and only from the CSR Protected List module in NVM. It can be written by the EMP and by the host only when in blank Flash programming mode.
Reserved	31:24	0x0	Reserved	Reserved.

10.2.1.7 Analyzer Registers

Registers used for the analyzer configuration.

10.2.1.7.1 L2 Tag Control - GL_SWT_L2TAGCTRL[n] (0x001C0A70 + 0x4*n, n=0..7)

Control register for L2 tags.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LENGTH	6:0	0x0	RW	Describes the number of bytes expected for this tag not including the Ethertype (2/4/6/8 if the <i>Long</i> bit is not set; any value otherwise). Length is in bytes (up to 126).
HAS_UP	7	0b	RW	Indicates that this tag has a <i>UP</i> field and this field should be taken into account for UP-to-TC translation if this is the first tag of the packet.
Reserved	8	0b	Reserved	Reserved.
ISVLAN	9	0b	RW	If this bit is set, then this tag is a VLAN tag. In this case, a tag with VLAN ID = 0b is treated as untagged (priority tagging) and the priority bits might be extracted to the Rx descriptor.
INNERUP	10	0b	RW	If this bit is set, the UP remapping is done on this field. Should be set only on one tag. If this bit is set, then <i>ISVLAN</i> should also be set.
OUTERUP	11	0b	RW	If this bit is set, then this is the tag on which the inner-to-outer UP remapping is applied. Should be set only in one tag.
LONG	12	0b	RW	Do not support insert of entire header. In this case, the length might be higher than 8 bytes.
Reserved	15:14	0x0	Reserved	Reserved.
ETHERTYPE	31:16	0x0	RW	The Ethertype identifying the L2 tag.

**10.2.1.7.2 L2 Tag - Enable - PRT_L2TAGSEN[PRT] (0x001C0B20 + 0x4*PRT, PRT=0...3)**

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
ENABLE	7:0	0x0	RW	Defines the L2 tags expected on this port.
Reserved	31:8	0x0	Reserved	Reserved.

Note: See the section about L2 tag handling for more detail.**10.2.1.8 Switch Registers**

Registers used to describe switch behavior.

10.2.1.8.1 Switching Table Default Action Enable Bitmap - GL_SWR_DEF_ACT_EN[n] (0x0026CFB8 + 0x4*n, n=0...1)

Switching table default action enable bitmap corresponding to GL_SWR_DEF_ACT.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DEF_ACT_EN_BITMAP	31:0	0x0	RW	Switching table default action enabling bitmap.

10.2.1.8.2 Switching Table Default Action - GL_SWR_DEF_ACT[n] (0x00270200 + 0x4*n, n=0...35)

For each switching table that doesn't hit for a packet, the default action takes place in case it's corresponding enable bit is set in GL_SWR_DEF_ACT_EN register.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DEF_ACTION	31:0	0x0	RW	32-bit switching action per table.

10.2.1.8.3 Port - TC Transmit UP Replacement - PRT_TCTUPR[n,PRT] (0x00044000 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

Defines the TC based Tx UP remapping.

This register must be programmed consistently with the programming made in the PRTDCB_TUP2TC register.



Note: The hardware default to this array must be set according to the hardware default of PRTDCB_TUP2TC.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
UP0	2:0	0x0	RW	Defines the UP to set if the UP in the packet is zero and the packet is sent through TC #n.
UP1	5:3	0x1	RW	Defines the UP to set if the UP in the packet is zero and the packet is sent through TC #n.
UP2	8:6	0x2	RW	Defines the UP to set if the UP in the packet is zero and the packet is sent through TC #n.
UP3	11:9	0x3	RW	Defines the UP to set if the UP in the packet is zero and the packet is sent through TC #n.
UP4	14:12	0x4	RW	Defines the UP to set if the UP in the packet is zero and the packet is sent through TC #n.
UP5	17:15	0x5	RW	Defines the UP to set if the UP in the packet is zero and the packet is sent through TC #n.
UP6	20:18	0x6	RW	Defines the UP to set if the UP in the packet is zero and the packet is sent through TC #n.
UP7	23:21	0x7	RW	Defines the UP to set if the UP in the packet is zero and the packet is sent through TC #n.
Reserved	31:24	0x0	Reserved	Reserved.

10.2.1.9 VSI Context

These registers create the VSI context. Unless otherwise stated, these registers are accessed by internal firmware. There are 384 VSI contexts shared between all PFs. All VSIs are accessible to all functions, but each function should access only the VSIs allocated to it.

10.2.1.9.1 VSI Tag Accept Register - VSI_TAR[VSI] (0x00042000 + 0x4*VSI, VSI=0...383)

Defines which tags can be accepted from the software device driver. This register impacts only tag insertion from the software device driver and not tag insertion by hardware.

Reset Source: PFR

NVM Load: No

Scope: VSI

Field	Bit(s)	Init.	Access Type	Description
ACCEPT TAGGED	9:0	0x0	RW	A bitmap describing if a packet with tag N is accepted. Note that the two first bits are mapped to the pre L2 and MAC headers.
Reserved	15:10	0x0	Reserved	Reserved.



Field	Bit(s)	Init.	Access Type	Description
ACCEPT UNTAGGED	25:16	0x0	RW	A bitmap that describes if a packet without tag N is accepted. If L2TAGCTRL.ISVLAN is set, also admits priority tagged packets (VLAN tag = 0). Note that the two first bits are mapped to the pre-L2 and MAC headers.
Reserved	31:26	0x0	Reserved	Reserved.

10.2.1.10 Interrupt Registers

10.2.1.10.1 Global Interrupt Control - GLINT_CTL (0x0003F800)

This register might be needed to be modified when operating in polling mode with no interrupts.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DIS_AUTOMASK_PFO	0	0b	RW	When DIS_AUTOMASK_PFO is cleared (default setting), the INTENA flag of interrupt zero of all PFs is auto-cleared following interrupt assertion from the ITR block to the PCIe block. When this flag is set to 1b, the INTENA flag is not auto-cleared and software should clear it as part of the interrupt routine.
DIS_AUTOMASK_VFO	1	0b	RW	When DIS_AUTOMASK_VFO is cleared (default setting), the INTENA flag of interrupt zero of all VFs is auto-cleared following interrupt assertion from the ITR block to the PCIe block. When this flag is set to 1b, the INTENA flag is not auto-cleared and software should clear it as part of the interrupt routine.
DIS_AUTOMASK_N	2	0b	RW	When DIS_AUTOMASK_N is cleared (default setting), the INTENA flag of all non-zero interrupts is auto-cleared following interrupt assertion from the ITR block to the PCIe block. When this flag is set to 1b, the INTENA flag is not auto-cleared and software should clear it as part of the interrupt routine.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.10.2 LAN Port MDIO Number - PFGEN_PORTMDIO_NUM[PF] (0x0003F100 + 0x4*PF, PF=0...15)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PORT_NUM	1:0	0x0	RW	Port Number. Indicates the LAN port connected to this function. 00b = Port 0. 01b = Port 1. 10b = Port 2. 11b = Port 3. This field must be identical to the PFGEN_PORTNUM register.
Reserved	3:2	0x0	Reserved	Reserved.
Reserved	31:5	0x0	Reserved	Reserved.



10.2.1.10.3 PF Interrupt Zero Cause - PFINT_ICR0[PF] (0x00038780 + 0x4*PF, PF=0...15)

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
INTEVENT	0	0b	RCW	Interrupt Event indication. This bit is set on assertion of any causes for this interrupt and cleared when the interrupt is asserted to the rate limit logic.
QUEUE_0	1	0b	RCW	Queue 0 interrupt for LAN transmit and receive queues.
QUEUE_1	2	0b	RCW	Queue 1 interrupt for LAN transmit and receive queues.
QUEUE_2	3	0b	RCW	Queue 2 interrupt for LAN transmit and receive queues.
QUEUE_3	4	0b	RCW	Queue 3 interrupt for LAN transmit and receive queues.
QUEUE_4	5	0b	RCW	Queue 4 interrupt for LAN transmit and receive queues.
QUEUE_5	6	0b	RCW	Queue 5 interrupt for LAN transmit and receive queues.
QUEUE_6	7	0b	RCW	Queue 6 interrupt for LAN transmit and receive queues.
QUEUE_7	8	0b	RCW	Queue 7 interrupt for LAN transmit and receive queues.
Reserved	15:9	0x0	Reserved	Reserved.
ECC_ERR	16	0b	RCW	Unrecoverable ECC Error. This bit is set when an unrecoverable error is detected in one of the device memories.
Reserved	18:17	0x0	Reserved	Reserved.
MAL_DETECT	19	0b	RCW	Malicious programming detected.
GRST	20	0b	RCW	Global Resets Requested (CORER, GLOBR or EMPR).
GPIO	22	0b	RCW	GPIO event indicates an event on any of the GPIO pins enabled for interrupt by the PFINT_GPIOCTL register. The GPIO state can be fetched on the GLGEN_GPIO_STAT register. The level transition that generates an interrupt is set for GPIO 'n' by the INT_MODE field in the matched GLGEN_GPIO_CTL[n] register.
TIMESYNC	23	0b	RCW	Any of the TimeSync interrupt causes as described in the interrupts section of the TimeSync (IEEE1588 and 802.1AS) section.
HMC_ERR	26	0b	RCW	HMC error as indicated in the PFHMC_ERRORINFO and PFCHMC_ERRORDATA registers.
Reserved	27	0b	Reserved	Reserved.
VFLR	29	0b	RCW	VFLR was initiated by one of the VFs of the PF. The PF should read GLGEN_VFLRSTAT getting an indication for the VF that generated the VFLR.
ADMINQ	30	0b	RCW	Send / receive admin queue interrupt indication.
SWINT	31	0b	RCW	Software Interrupt Indication.

10.2.1.10.4 PF Interrupt Zero Cause Enablement - PFINT_ICR0_ENA[PF] (0x00038800 + 0x4*PF, PF=0...15)

Reset Source: CORER

NVM Load: No

Scope: PF



Field	Bit(s)	Init.	Access Type	Description
Reserved	15:0	0x0	Reserved	Reserved.
ECC_ERR	16	0b	RW	Enable this interrupt at 1b.
Reserved	18:17	0x0	Reserved	Reserved.
MAL_DETECT	19	0b	RW	Enable this interrupt at 1b.
GRST	20	0b	RW	Enable this interrupt at 1b.
GPIO	22	0b	RW	Enable this interrupt at 1b.
TIMESYNC	23	0b	RW	Enable this interrupt at 1b.
HMC_ERR	26	0b	RW	Enable this interrupt at 1b.
Reserved	27	0b	Reserved	Reserved.
VFLR	29	0b	RW	Enable this interrupt at 1b.
ADMINQ	30	0b	RW	Enable this interrupt at 1b.
Reserved	31	0b	RW	Reserved.

10.2.1.10.5 PF Interrupt Zero Dynamic Control - PFINT_DYN_CTL0[PF] (0x00038480 + 0x4*PF, PF=0...15)

In case of MSI or legacy INTA mode of operation, interrupt zero is the only valid interrupt.

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
INTENA	0	0b	RW	Interrupt Enable. At 1b, interrupt is enabled. At 0b, interrupt is disabled. This bit is meaningful only if the INTENA_MSK flag in this register is not set.
CLEARPBA	1	0b	RW1C	When setting this bit, the matched PBA bit is cleared. This bit is auto-cleared by hardware.
SWINT_TRIG	2	0b	RW1C	Trigger Software Interrupt. When the bit is set, a software interrupt is triggered. This bit is auto-cleared by hardware.
ITR_INDX	4:3	0x0	RW1C	This field defines the ITR index to be updated as follows. It is auto-cleared by hardware. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = No ITR update.
INTERVAL	16:5	0x0	RW1C	The interval for the ITR defined by the ITR_INDX in this register. It is auto-cleared by hardware.
Reserved	23:17	0x0	Reserved	Reserved.
SW_ITR_INDX_ENA	24	0b	RW1C	This flag enables the programming of SW_ITR_INDX in this register. This flag is auto-cleared by hardware.
SW_ITR_INDX	26:25	0x0	RW	ITR index of the software interrupt. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR. When programming this field, the SW_ITR_INDX_ENA flag in this register should be set as well.



Field	Bit(s)	Init.	Access Type	Description
Reserved	30:27	0x0	Reserved	Reserved.
INTENA_MSK	31	0b	RW1C	When the INTENA_MSK bit is set, the INTENA setting does not impact the device setting. This bit is auto-cleared by hardware.

10.2.1.10.6 PF Interrupt Zero Static Control - PFINT_STAT_CTL0[PF] (0x00038400 + 0x4*PF, PF=0...15)

In case of MSI or legacy INTA mode of operation, interrupt zero is the only valid interrupt.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
Reserved	1:0	0x0	Reserved	Reserved.
OTHER_ITR_INDXX	3:2	0x0	RW	ITR index of the other interrupt causes. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR.
Reserved	31:4	0x0	Reserved	Reserved.

10.2.1.10.7 PF Interrupt Zero Linked List - PFINT_LNKLST0[PF] (0x00038500 + 0x4*PF, PF=0...15)

In case of MSI or legacy INTA mode of operation, interrupt zero is the only valid interrupt.

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FIRSTQ_INDXX	10:0	0x7FF	RW	First Queue Index in the MSI-X Cause List. Transmit and receive queue indexes are within the PF space. CEQ indexes are within the function's space. Setting the index to 0x7FF points to an empty linked list (might be useful for the other cause interrupt).
FIRSTQ_TYPE	12:11	0x0	RW	First Queue Type. It can be one of the following: 00b = Receive queues. 01b = Transmit queues. 10b = Reserved. 11b = Reserved.
Reserved	31:13	0x0	Reserved	Reserved.



10.2.1.10.8 PF Interrupt N Dynamic Control - PFINT_DYN_CTLN[INTPF] (0x00034800 + 0x4*INTPF, INTPF=0...511)

Register index 'n' relates to interrupt 'n+1' while interrupt zero is controlled by the PFINT_DYN_CTL0 register.

In case of MSI or legacy INTA only interrupt zero is valid so all these registers do not impact device functionality.

Reset Source: PFR

NVM Load: No

Scope: INTPF

Field	Bit(s)	Init.	Access Type	Description
INTENA	0	0b	RW	Interrupt Enable. At 1b the interrupt is enabled. At 0b the interrupt is disabled. This bit is meaningful only if the INTENA_MSK flag in this register is not set.
CLEARPBA	1	0b	RW1C	When setting this bit, the matched PBA bit is cleared. This bit is auto-cleared by hardware.
SWINT_TRIG	2	0b	RW1C	Trigger Software Interrupt. When the bit is set, a software interrupt is triggered. This bit is auto-cleared by hardware.
ITR_INDX	4:3	0x0	RW1C	This field defines the ITR Index to be updated as follows. It is auto-cleared by hardware. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = No ITR update.
INTERVAL	16:5	0x0	RW1C	The interval for the ITR defined by the ITR_INDX in this register. It is auto-cleared by hardware.
Reserved	23:17	0x0	Reserved	Reserved.
SW_ITR_INDX_ENA	24	0b	RW1C	This flag enables the programming of the SW_ITR_INDX in this register. This flag is auto-cleared by hardware.
SW_ITR_INDX	26:25	0x0	RW	ITR Index of the Software Interrupt. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR. When programming this field, the SW_ITR_INDX_ENA flag in this register should be set as well.
Reserved	30:27	0x0	Reserved	Reserved.
INTENA_MSK	31	0b	RW1C	When the INTENA_MSK bit is set, then the INTENA setting does not impact the device setting. This bit is auto-cleared by hardware.



10.2.1.10.9 PF Interrupt N Linked List - PFINT_LNKLSTN[INTPF] (0x00035000 + 0x4*INTPF, INTPF=0...511)

Register index 'n' relates to interrupt 'n+1' while interrupt zero is controlled by the PFINT_LNKLST0 register.

In case of MSI or legacy INTA only interrupt zero is valid so all these registers do not impact device functionality.

Reset Source: PFR

NVM Load: No

Scope: INTPF

Field	Bit(s)	Init.	Access Type	Description
FIRSTQ_INDX	10:0	0x7FF	RW	First Queue Index in the MSI-X Cause List. Transmit and receive queue indexes are within the PF space. CEQ indexes are within the function's space. Setting the index to 0x7FF points to an empty linked list (might be useful for the other cause interrupt).
FIRSTQ_TYPE	12:11	0x0	RW	First Queue Type. It can be one of the following: 00b = Receive queues. 01b = Transmit queues. 10b = Reserved. 11b = Reserved.
Reserved	31:13	0x0	Reserved	Reserved.

10.2.1.10.10 PF Interrupt Throttling for Interrupt Zero - PFINT_ITR0[n,PF] (0x00038000 + 0x80*n + 0x4*PF, n=0...2, PF=0...15)

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
INTERVAL	11:0	0x0	RW	ITR 'n' interval, while 'n' is the register index = 0,1,2 for the three ITRs per interrupt. It is defined in 2 μ s units enabling interval range from zero to 8160 μ s (0xFF0). Setting INTERVAL to zero enables an immediate interrupt. This register can also be programmed by setting the <i>INTERVAL</i> field in the matched xxINT_DYN_CTLx register.
Reserved	31:12	0x0	Reserved	Reserved.



10.2.1.10.11 PF Interrupt Throttling for Interrupt N - PFINT_ITRN[n,INTPF] (0x00030000 + 0x800*n + 0x4*INTPF, n=0...2, INTPF=0...511)

Register index 'n' relates to interrupt 'n+1' while interrupt zero is controlled by the PFINT_ITR0 register.

In case of MSI or legacy INTA only interrupt zero is valid so all these registers do not impact the device functionality.

Reset Source: PFR

NVM Load: No

Scope: INTPF

Field	Bit(s)	Init.	Access Type	Description
INTERVAL	11:0	0x0	RW	ITR 'n' interval, while 'n' is the register index = 0,1,2 for the three ITRs per interrupt. It is defined in 2 μs units enabling interval range from zero to 8160 μs (0xFF0). Setting the INTERVAL to zero enables an immediate interrupt. This register can also be programmed by setting the INTERVAL field in the matched xxINT_DYN_CTLx register.
Reserved	31:12	0x0	Reserved	Reserved.

10.2.1.10.12 PF Interrupt Zero Rate Limit - PFINT_RATE0[PF] (0x00038580 + 0x4*PF, PF=0...15)

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
INTERVAL	5:0	0x0	RW	Time interval defined in 4 μs units between consecutive credit incremental. When the interrupt rate limit is enabled by the INTRL_ENA flag in this register, the INTERVAL must be greater than zero. Also, for accurate rate limit, INTERVAL must be smaller than 0x3C (up to 236 μs).
INTRL_ENA	6	0b	RW	Enable interrupt rate limit on this interrupt vector.
Reserved	31:7	0x0	Reserved	Reserved.



10.2.1.10.13 PF Interrupt N Rate Limit - PFINT_RATE[NINTPF] (0x00035800 + 0x4*NINTPF, NINTPF=0...511)

Register index 'n' relates to interrupt 'n+1' while interrupt zero is controlled by the PFINT_RATE0 register.

In case of MSI or legacy INTA only interrupt zero is valid so all these registers do not impact device functionality.

Reset Source: PFR

NVM Load: No

Scope: NINTPF

Field	Bit(s)	Init.	Access Type	Description
INTERVAL	5:0	0x0	RW	Time interval defined in 4 μ s units between consecutive credit incremental. When the interrupt rate limit is enabled by the INTRL_ENA flag in this register, the INTERVAL must be greater than zero. Also, for accurate rate limit, the INTERVAL must be smaller than 0x3C (up to 236 μ s).
INTRL_ENA	6	0b	RW	Enable interrupt rate limit on this interrupt vector.
Reserved	31:7	0x0	Reserved	Reserved.

10.2.1.10.14 Receive Queue Interrupt Cause Control - QINT_RQCTL[Q] (0x0003A000 + 0x4*Q, Q=0...1535)

Reset Source: CORER

NVM Load: No

Scope: Q

Field	Bit(s)	Init.	Access Type	Description
MSIX_INDX	7:0	0x0	RW	MSI-X vector index within the function space. Software should set MSIX_INDX to values in the range of allocated interrupt vectors to the function.
Reserved	10:8	0x0	Reserved	Reserved.
ITR_INDX	12:11	0x0	RW	ITR Index of the interrupt cause. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR.
MSIX0_INDX	15:13	0x0	RW	The index of the QUEUE_x bits in the ICRO register ('x' = 0,...7) that is set as a result of an event on the queue. This field is relevant only if the MSIX_INDX equals to zero.
NEXTQ_INDX	26:16	0x0	RW	Next Queue Index in the MSI-X Cause List. Transmit and receive queue indexes are within the PF space. CEQ indexes are within the function's space. Setting the index to 0x7FF is a NULL pointer indicating the end of the linked list.
NEXTQ_TYPE	28:27	0x0	RW	Next Queue Type. It can be one of the following: 00b = Receive queues. 01b = Transmit queues. 10b = Reserved. 11b = Reserved.
Reserved	29	0b	Reserved	Reserved.



Field	Bit(s)	Init.	Access Type	Description
CAUSE_ENA	30	0b	RW	Enable interrupt by this queue. When CAUSE_ENA is cleared, interrupts are not generated by the queue. The queue remains in the interrupt linked list and is processed at ITR expiration.
INTEVENT	31	0b	RO	Interrupt Event Indication. It is triggered by the specific event on the queue and cleared when the interrupt is acknowledged by the interrupt signal logic.

10.2.1.10.15 Transmit Queue Interrupt Cause Control - QINT_TQCTL[Q] (0x0003C000 + 0x4*Q, Q=0...1535)

Reset Source: CORER

NVM Load: No

Scope: Q

Field	Bit(s)	Init.	Access Type	Description
MSIX_INDx	7:0	0x0	RW	MSI-X vector index within the function space. Software should set the MSIX_INDx to values in the range of allocated interrupt vectors to the function.
Reserved	10:8	0x0	Reserved	Reserved.
ITR_INDx	12:11	0x0	RW	ITR Index of the interrupt cause. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR.
MSIX0_INDx	15:13	0x0	RW	The index of the QUEUE_x bits in the ICR0 register ('x' = 0,...7) that is set as a result of an event on the queue. This field is relevant only if the MSIX_INDx equals to zero.
NEXTQ_INDx	26:16	0x0	RW	Next Queue Index in the MSI-X Cause List. Transmit and receive queue indexes are within the PF space. CEQ indexes are within the function's space. Setting the index to 0x7FF is a NULL pointer indicating the end of the linked list.
NEXTQ_TYPE	28:27	0x0	RW	Next Queue Type. It can be one of the following: 00b = Receive queues. 01b = Transmit queues. 10b = Reserved. 11b = Reserved.
Reserved	29	0b	Reserved	Reserved.
CAUSE_ENA	30	0b	RW	Enable interrupt by this queue. When CAUSE_ENA is cleared, interrupts are not generated by the queue. The queue remains in the interrupt linked list and is processed at ITR expiration.
INTEVENT	31	0b	RO	Interrupt Event Indication. It is triggered by the specific event on the queue and cleared when the interrupt is acknowledged by the interrupt signal logic.



**10.2.1.10.16PF General Purpose IO Interrupt Enablement - PFINT_GPIO_ENA[PF]
(0x00088080 + 0x4*PF, PF=0...15)**

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
GPIO0_ENA	0	0b	RW	Enable interrupt on GPIO 0.
GPIO1_ENA	1	0b	RW	Enable interrupt on GPIO 1.
GPIO2_ENA	2	0b	RW	Enable interrupt on GPIO 2.
GPIO3_ENA	3	0b	RW	Enable interrupt on GPIO 3.
GPIO4_ENA	4	0b	RW	Enable interrupt on GPIO 4.
GPIO5_ENA	5	0b	RW	Enable interrupt on GPIO 5.
GPIO6_ENA	6	0b	RW	Enable interrupt on GPIO 6.
GPIO7_ENA	7	0b	RW	Enable interrupt on GPIO 7.
GPIO8_ENA	8	0b	RW	Enable interrupt on GPIO 8.
GPIO9_ENA	9	0b	RW	Enable interrupt on GPIO 9.
GPIO10_ENA	10	0b	RW	Enable interrupt on GPIO 10.
GPIO11_ENA	11	0b	RW	Enable interrupt on GPIO 11.
GPIO12_ENA	12	0b	RW	Enable interrupt on GPIO 12.
GPIO13_ENA	13	0b	RW	Enable interrupt on GPIO 13.
GPIO14_ENA	14	0b	RW	Enable interrupt on GPIO 14.
GPIO15_ENA	15	0b	RW	Enable interrupt on GPIO 15.
GPIO16_ENA	16	0b	RW	Enable interrupt on GPIO 16.
GPIO17_ENA	17	0b	RW	Enable interrupt on GPIO 17.
GPIO18_ENA	18	0b	RW	Enable interrupt on GPIO 18.
GPIO19_ENA	19	0b	RW	Enable interrupt on GPIO 19.
GPIO20_ENA	20	0b	RW	Enable interrupt on GPIO 20.
GPIO21_ENA	21	0b	RW	Enable interrupt on GPIO 21.
GPIO22_ENA	22	0b	RW	Enable interrupt on GPIO 22.
GPIO23_ENA	23	0b	RW	Enable interrupt on GPIO 23.
GPIO24_ENA	24	0b	RW	Enable interrupt on GPIO 24.
GPIO25_ENA	25	0b	RW	Enable interrupt on GPIO 25.
GPIO26_ENA	26	0b	RW	Enable interrupt on GPIO 26.
GPIO27_ENA	27	0b	RW	Enable interrupt on GPIO 27.
GPIO28_ENA	28	0b	RW	Enable interrupt on GPIO 28.
GPIO29_ENA	29	0b	RW	Enable interrupt on GPIO 29.
Reserved	31:30	0x0	Reserved	Reserved.



10.2.1.10.17 EMP General Purpose IO Interrupt Enablement - EMPINT_GPIO_ENA (0x00088188)

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GPIO0_ENA	0	0b	RW	Enable interrupt on GPIO 0.
GPIO1_ENA	1	0b	RW	Enable interrupt on GPIO 1.
GPIO2_ENA	2	0b	RW	Enable interrupt on GPIO 2.
GPIO3_ENA	3	0b	RW	Enable interrupt on GPIO 3.
GPIO4_ENA	4	0b	RW	Enable interrupt on GPIO 4.
GPIO5_ENA	5	0b	RW	Enable interrupt on GPIO 5.
GPIO6_ENA	6	0b	RW	Enable interrupt on GPIO 6.
GPIO7_ENA	7	0b	RW	Enable interrupt on GPIO 7.
GPIO8_ENA	8	0b	RW	Enable interrupt on GPIO 8.
GPIO9_ENA	9	0b	RW	Enable interrupt on GPIO 9.
GPIO10_ENA	10	0b	RW	Enable interrupt on GPIO 10.
GPIO11_ENA	11	0b	RW	Enable interrupt on GPIO 11.
GPIO12_ENA	12	0b	RW	Enable interrupt on GPIO 12.
GPIO13_ENA	13	0b	RW	Enable interrupt on GPIO 13.
GPIO14_ENA	14	0b	RW	Enable interrupt on GPIO 14.
GPIO15_ENA	15	0b	RW	Enable interrupt on GPIO 15.
GPIO16_ENA	16	0b	RW	Enable interrupt on GPIO 16.
GPIO17_ENA	17	0b	RW	Enable interrupt on GPIO 17.
GPIO18_ENA	18	0b	RW	Enable interrupt on GPIO 18.
GPIO19_ENA	19	0b	RW	Enable interrupt on GPIO 19.
GPIO20_ENA	20	0b	RW	Enable interrupt on GPIO 20.
GPIO21_ENA	21	0b	RW	Enable interrupt on GPIO 21.
GPIO22_ENA	22	0b	RW	Enable interrupt on GPIO 22.
GPIO23_ENA	23	0b	RW	Enable interrupt on GPIO 23.
GPIO24_ENA	24	0b	RW	Enable interrupt on GPIO 24.
GPIO25_ENA	25	0b	RW	Enable interrupt on GPIO 25.
GPIO26_ENA	26	0b	RW	Enable interrupt on GPIO 26.
GPIO27_ENA	27	0b	RW	Enable interrupt on GPIO 27.
GPIO28_ENA	28	0b	RW	Enable interrupt on GPIO 28.
GPIO29_ENA	29	0b	RW	Enable interrupt on GPIO 29.
Reserved	31:30	0x0	Reserved	Reserved.



10.2.1.10.18 VF Interrupt Zero Cause - VFINT_ICR0[VF] (0x0002BC00 + 0x4*VF, VF=0...127)

Reset Source: CORER

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
INTEVENT	0	0b	RCW	Interrupt Event Indication. This bit is set on assertion of any causes for this interrupt and cleared when the interrupt is asserted to the rate limit logic.
QUEUE_0	1	0b	RCW	Queue 0 interrupt for LAN transmit and receive queues.
QUEUE_1	2	0b	RCW	Queue 1 interrupt for LAN transmit and receive queues.
QUEUE_2	3	0b	RCW	Queue 2 interrupt for LAN transmit and receive queues.
QUEUE_3	4	0b	RCW	Queue 3 interrupt for LAN transmit and receive queues.
Reserved	24:5	0x0	Reserved	Reserved.
Reserved	29:26	0x0	Reserved	Reserved.
ADMINQ	30	0b	RCW	Send / Receive Admin Queue Interrupt Indication.
SWINT	31	0b	RCW	Software Interrupt Indication.

10.2.1.10.19 VF Interrupt Zero Cause Enablement - VFINT_ICR0_ENA[VF] (0x0002C000 + 0x4*VF, VF=0...127)

Reset Source: CORER

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
Reserved	24:0	0x0	Reserved	Reserved.
Reserved	29:26	0x0	Reserved	Reserved.
ADMINQ	30	0b	RW	Enable this interrupt at 1b.
Reserved	31	0b	RW	Reserved.

10.2.1.10.20 VF Interrupt Zero Dynamic Control - VFINT_DYN_CTL0[VF] (0x0002A400 + 0x4*VF, VF=0...127)

Reset Source: VFR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
INTENA	0	0b	RW	Interrupt Enable. At 1b, the interrupt is enabled. At 0b, the interrupt is disabled. This bit is meaningful only if the INTENA_MSK flag in this register is not set.
CLEARPBA	1	0b	RW1C	When setting this bit, the matched PBA bit is cleared. This bit is auto-cleared by hardware.



Field	Bit(s)	Init.	Access Type	Description
SWINT_TRIG	2	0b	RW1C	Trigger Software Interrupt. When this bit is set, a software interrupt is triggered. This bit is auto-cleared by hardware.
ITR_INDX	4:3	0x0	RW1C	This field defines the ITR index to be updated as follows. It is auto-cleared by hardware. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR update.
INTERVAL	16:5	0x0	RW1C	The interval for the ITR defined by ITR_INDX in this register. It is auto-cleared by hardware.
Reserved	23:17	0x0	Reserved	Reserved.
SW_ITR_INDX_ENA	24	0b	RW1C	This flag enables the programming of SW_ITR_INDX in this register. This flag is auto-cleared by hardware.
SW_ITR_INDX	26:25	0x0	RW	ITR Index of Software Interrupt. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR. When programming this field, the SW_ITR_INDX_ENA flag in this register should be set as well.
Reserved	30:27	0x0	Reserved	Reserved.
INTENA_MSK	31	0b	RW1C	When the INTENA_MSK bit is set, then the INTENA setting does not impact the device setting. This bit is auto-cleared by hardware.

10.2.1.10.21 VF Interrupt Zero Static Control - VFINT_STAT_CTL0[VF] (0x0002A000 + 0x4*VF, VF=0...127)

In case of MSI or legacy INTA mode of operation interrupt zero is the only valid interrupt.

Reset Source: CORER

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
Reserved	1:0	0x0	Reserved	Reserved.
OTHER_ITR_INDX	3:2	0x0	RW	ITR Index of other interrupt causes. 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR.
Reserved	31:4	0x0	Reserved	Reserved.



10.2.1.10.22 Protected VF Interrupt Zero Linked List - VPINT_LNKLST0[VF] (0x0002A800 + 0x4*VF, VF=0...127)

In case of MSI or legacy INTA mode of operation interrupt zero is the only valid interrupt.

Reset Source: VFR

NVM Load: No

Scope: VP

Field	Bit(s)	Init.	Access Type	Description
FIRSTQ_INDX	10:0	0x7FF	RW	First Queue Index in the MSI-X Cause List. Transmit and receive queue indexes are within the PF space. CEQ indexes are within the function's space. Setting the index to 0x7FF points to an empty linked list (might be useful for other cause interrupt).
FIRSTQ_TYPE	12:11	0x0	RW	First Queue Type. It can be one of the following. 00b = Receive queues. 01b = Transmit queues. 10b = Reserved. 11b = Reserved.
Reserved	31:13	0x0	Reserved	Reserved.

10.2.1.10.23 VF Interrupt N Dynamic Control - VFINT_DYN_CTLN[INTVF] (0x00024800 + 0x4*INTVF, INTVF=0...511)

Register index 'n' relates to interrupt 'n+1' while interrupt zero is controlled by the VFINT_DYN_CTL0 register.

Reset Source: VFR

NVM Load: No

Scope: INTVF

Field	Bit(s)	Init.	Access Type	Description
INTENA	0	0b	RW	Interrupt Enable. At 1b the interrupt is enabled. At 0b the interrupt is disabled. This bit is meaningful only if the INTENA_MSK flag in this register is not set.
CLEARPBA	1	0b	RW1C	When setting this bit, the matched PBA bit is cleared. This bit is auto-cleared by hardware.
SWINT_TRIG	2	0b	RW1C	Trigger Software Interrupt. When the bit is set, a software interrupt is triggered. This bit is auto-cleared by hardware.
ITR_INDX	4:3	0x0	RW1C	This field defines the ITR index to be updated as follows. It is auto-cleared by hardware: 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR update.
INTERVAL	16:5	0x0	RW1C	The interval for the ITR defined by the ITR_INDX in this register. It is auto-cleared by hardware.
Reserved	23:17	0x0	Reserved	Reserved.
SW_ITR_INDX_ENA	24	0b	RW1C	This flag enables the programming of SW_ITR_INDX in this register. This flag is auto cleared by hardware.



Field	Bit(s)	Init.	Access Type	Description
SW_ITR_IND _X	26:25	0x0	RW	ITR Index of Software Interrupt: 00b = ITR0. 01b = ITR1. 10b = ITR2. 11b = NoITR. When programming this field, the SW_ITR_IND _X _ENA flag in this register should be set as well.
Reserved	30:27	0x0	Reserved	Reserved.
INTENA_MSK	31	0b	RW1C	When the INTENA_MSK bit is set, then the INTENA setting does not impact the device setting. This bit is auto-cleared by hardware.

10.2.1.10.24 Protected VF Interrupt N Linked List - VPINT_LNKLSTN[INTVF] (0x00025000 + 0x4*INTVF, INTVF=0...511)

Register index 'n' relates to interrupt 'n+1' while interrupt zero is controlled by the VPINT_LNKLST0 register.

Reset Source: VFR

NVM Load: No

Scope: INTVP

Field	Bit(s)	Init.	Access Type	Description
FIRSTQ_IND _X	10:0	0x7FF	RW	First Queue Index in the MSI-X Cause List. Transmit and receive queue indexes are within the PF space. CEQ indexes are within the function's space. Setting the index to 0x7FF points to an empty linked list (might be useful for other cause interrupt).
FIRSTQ_TYPE	12:11	0x0	RW	First Queue Type. It can be one of the following. 00b = Receive queues. 01b = Transmit queues. 10b = Reserved. 11b = Reserved.
Reserved	31:13	0x0	Reserved	Reserved.

10.2.1.10.25 VF Interrupt Throttling for Interrupt Zero - VFINT_ITR0[n,VF] (0x00028000 + 0x400*n + 0x4*VF, n=0...2, VF=0...127) itr_regs

Reset Source: VFR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
INTERVAL	11:0	0x0	RW	ITR 'n' interval, while 'n' is the register index = 0,1,2 for the three ITRs per interrupt. It is defined in 2 μs units enabling an interval range from zero to 8160 μs (0xFF0). Setting <i>INTERVAL</i> to zero enables an immediate interrupt. This register can also be programmed by setting the <i>INTERVAL</i> field in the matched xxINT_DYN_CTLx register.
Reserved	31:12	0x0	Reserved	Reserved.



10.2.1.10.26 VF Interrupt Throttling for Interrupt N - VFINT_ITRN[n,INTVF] (0x00020000 + 0x800*n + 0x4*INTVF, n=0...2, INTVF=0...511)

Register index 'n' relates to interrupt 'n+1' while interrupt zero is controlled by the VFINT_ITR0 register.

Reset Source: VFR

NVM Load: No

Scope: INTVF

Field	Bit(s)	Init.	Access Type	Description
INTERVAL	11:0	0x0	RW	ITR 'n' interval, while 'n' is the register index = 0,1,2 for the three ITRs per interrupt. It is defined in 2 μ s units enabling an interval range from zero to 8160 μ s (0xFF0). Setting <i>INTERVAL</i> to zero enables an immediate interrupt. This register can also be programmed by setting the <i>INTERVAL</i> field in the matched xxINT_DYN_CTLx register.
Reserved	31:12	0x0	Reserved	Reserved.

10.2.1.10.27 Protected VF Interrupt Zero Rate Limit - VPINT_RATE0[VF] (0x0002AC00 + 0x4*VF, VF=0...127)

This register affects the VF but exposed only to the parent PF.

Reset Source: VFR

NVM Load: No

Scope: VP

Field	Bit(s)	Init.	Access Type	Description
INTERVAL	5:0	0x0	RW	Time interval defined in 4 μ s units between consecutive credit incremental. When the interrupt rate limit is enabled by the INTRL_ENA flag in this register, <i>INTERVAL</i> must be greater than zero. And for accurate rate limit, <i>INTERVAL</i> must be smaller than 0x3C (up to 236 μ s).
INTRL_ENA	6	0b	RW	Enable interrupt rate limit on this interrupt vector.
Reserved	31:7	0x0	Reserved	Reserved.

10.2.1.10.28 Protected VF Interrupt N Rate Limit - VPINT_RATEN[INTVF] (0x00025800 + 0x4*INTVF, INTVF=0...511)

These registers affect the VF but exposed only to the parent PF.

Register index 'n' relates to interrupt 'n+1' while interrupt zero is controlled by the VFINT_RATE0 register.

Reset Source: VFR.

NVM Load: No

Scope: INTVP



Field	Bit(s)	Init.	Access Type	Description
INTERVAL	5:0	0x0	RW	Time interval defined in 4 μ s units between consecutive credit incremental. When the interrupt rate limit is enabled by the INTRL_ENA flag in this register, <i>INTERVAL</i> must be greater than zero. And for accurate rate limit, <i>INTERVAL</i> must be smaller than 0x3C (up to 236 μ s).
INTRL_ENA	6	0b	RW	Enable interrupt rate limit on this interrupt vector.
Reserved	31:7	0x0	Reserved	Reserved.

10.2.1.11 Virtualization PF Registers

10.2.1.11.1 Malicious Driver Detected on Rx - VP_MDET_RX[VF] (0x0012A000 + 0x4*VF, VF=0...127)

This register records a malicious event detected on the Rx queues. Once read, the software device driver must write 0xFFFF to clear.

Reset Source: CORER

NVM Load: No

Scope: VP

Field	Bit(s)	Init.	Access Type	Description
VALID	0	0b	RW1C	A malicious event has been detected on this function.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.11.2 Malicious Driver Rx Event Details - GL_MDET_RX (0x0012A510)

This register records the details of the first Rx event detected.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FUNCTION	7:0	0x0	RW1C	The function that triggered the event.
EVENT	16:8	0x0	RW1C	ID of the event that has been recorded.
QUEUE	30:17	0x0	RW1C	Queue ID on which the event was detected.
VALID	31	0b	RW1C	Indicates that an event has been captured.



10.2.1.11.3 Malicious Driver Detected on Rx - PF_MDET_RX[PF] (0x0012A400 + 0x4*PF, PF=0...15)

This register records a malicious event detected on the Rx queues. Once read, the software device driver must write 0xFFFF to clear.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
VALID	0	0b	RW1C	A malicious event has been detected on this function.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.11.4 Malicious Driver Rx Checks Enabled - GL_MDCK_RX (0x0012A50C)

This register selects which Rx checks are enabled. For a list of the defined bits see the virtualization section.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DESC_ADDR	0	1b	RW	Enable Malicious Event. Descriptor fetch failed. For proper operation, this flag must be active.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.11.5 Malicious Driver Detected on Tx - VP_MDET_TX[VF] (0x000E6000 + 0x4*VF, VF=0...127)

This register records a malicious event detected on the Tx queues. Once read, the software device driver must write 0xFFFF to clear.

Reset Source: CORER

NVM Load: No

Scope: VP

Field	Bit(s)	Init.	Access Type	Description
VALID	0	0b	RW1C	A malicious event has been detected on this function.
Reserved	31:1	0x0	Reserved	Reserved.

**10.2.1.11.6 Malicious Driver Tx Event Details - GL_MDET_TX (0x000E6480)**

This register records the details of the first Tx event detected.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
QUEUE	11:0	0x0	RW1C	Absolute queue ID on which the event was detected.
VF_NUM	20:12	0x0	RW1C	Absolute VF number on which the event was detected.
PF_NUM	24:21	0x0	RW1C	PF / parent PF number on which the event was detected.
Reserved	30	0b	Reserved	Reserved.
VALID	31	0b	RW1C	Indicates that an event has been captured.

10.2.1.11.7 Malicious Driver Detected on Tx - PF_MDET_TX[PF] (0x000E6400 + 0x4*PF, PF=0...15)

This register records a malicious event detected on the Tx queues. Once read, the software device driver must write 0xFFFF to clear.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
VALID	0	0b	RW1C	A malicious event has been detected on this function.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.11.8 Malicious Driver Tx Command Checks Enable - GL_MDCK_TCMD (0x000E648C)

This register selects which Tx command checks are enabled. For a list of the defined bits, see the virtualization section.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DESC_ADDR	0	1b	RW	Enable malicious event. Descriptor fetch failed.
Reserved	1	0b	Reserved	Reserved.
MAX_BUFF	2	1b	RW	Enable malicious event. Single send packet or large send segment is spread on more than eight data descriptors.
MAX_HEAD	3	1b	RW	Enable malicious event. LSO with header spanning more than 3 buffers.
NO_HEAD	4	1b	RW	Enable malicious event. Zero header length when large send offload enabled.



Field	Bit(s)	Init.	Access Type	Description
WRONG_SIZE	5	1b	RW	Enable malicious event. A single send packet or large send segment that is not between min / max sizes defined in the GLTLAN_MIN_MAX_PKT register.
Reserved	6	1b	Reserved	
ENDLESS_TX	7	1b	RW	Enable malicious event. Endless transmit ring. When this flag is cleared, endless transmit ring option is enabled. When this flag is set, endless transmit ring (a tail update larger than the ring size) is considered a malicious event.
BAD_LSO_LEN	8	1b	RW	Enable malicious event. LSO PKT_LEN is not equal to the sum of buffers.
BAD_LSO_MSS	9	1b	RW	Enable malicious event. LSO with MSS is within the range defined by the GLTLAN_MIN_MAX_MSS register. Functionality of this flag is enabled by bit 13.
Reserved	11:10	0x0	Reserved	Reserved.
M_CONTEXTS	12	1b	RW	Enable malicious event. 7 or more consecutive non-data descriptors are fetched in a transmit queue. Functionality of this flag is a subset of bit 13.
BAD_DESC_SEQUENCE	13	1b	RW	Enable malicious event. Bad descriptors sequence. Include. MSS min/max, max number of non-data descriptors, DIF/DIX descriptors (enabled by bit 16) and flex descriptors (enabled by bit 17).
BAD_FC_FD_DESC	14	1b	RW	Enable malicious event. Illegal use FD filter descriptors. Activated on Tx queues that are not enabled for FD.
NO_PACKET	15	1b	RW	Enable malicious event. Tail update that does not contain at least one full packet.
DIS_DIF_DIX	16	1b	RW	Enable malicious event. DIF/DIX descriptors are considered malicious descriptors.
DIS_FLEX	17	1b	RW	Enable malicious event. Flexible descriptors are considered malicious descriptors.
ZERO_BSIZE	18	1b	RW	Enable malicious event. A descriptor with a zero value in BSIZE.
Reserved	31_19	0x0	Reserved	Reserved.

10.2.1.11.9 Malicious Driver Tx Data Checks Enable - GL_MDCK_TDAT (0x000442F4)

This register selects which Tx data checks are enabled. For a list of the defined bits, see the virtualization section.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
Reserved	1:0	1b	RW	Reserved.
MAL_LENGTH_DIS	2	0b	RW	Disable malicious detection of a bad length in the transmit descriptor.
MAL_CMD_DIS	3	0b	RW	Disable malicious detection of a bad combination of commands in the transmit descriptor.
Reserved	31:4	0x0	Reserved	Reserved.



10.2.1.11.10PF Resources Allocation - PF_VT_PFALLOC[PF] (0x001C0500 + 0x4*PF, PF=0...15)

This register indicates the VFs allocated to the each PF.

Note that the VFs can be allocated to LAN ports in granularity of 16 VFs. Under this constraint, these VFs can be distributed arbitrarily among the different PFs.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FIRSTVF	7:0	0x0	RW	The first VF allocated to this PF. Valid only if the <i>VALID</i> flag is set. Valid values are 0-127.
LASTVF	15:8	0x0	RW	The last VF allocated to this PF. Valid only if the <i>VALID</i> flag is set. Valid values are 0-127.
Reserved	30:16	0x0	Reserved	Reserved.
VALID	31	0b	RW	The <i>FIRSTVF</i> and <i>LASTVF</i> fields in this register are valid. If cleared, no VFs are allocated to this PF. If cleared, the SR-IOV capability should not be exposed for this PF.

10.2.1.12 DCB Registers

10.2.1.12.1 Global DCB General Control - GLDCB_GENC (0x00083044)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PCIRTT	15:0	0x009C	RW	PCIe Round Trip Time. It is expressed in 16 bytes units. Default is 2 μ s PCIe round trip time assuming 10 GbE links (no optimization done for lower links speeds).
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.12.2 Port DCB General Status - PRTDCB_GENS[PRT] (0x00083020 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: No

Scope: PRT



Field	Bit(s)	Init.	Access Type	Description
DCBX_STATUS	2:0	0x0	RW	DCBX Status. 0x0 = NOT_STARTED. 0x1 = IN_PROGRESS. 0x2 = DONE. 0x3 = MULTIPLE_PEERS. 0x4-0x6 = Reserved. 0x7 = DISABLED.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.12.3 Port DCB General Control - PRTDCB_GENC[PRT] (0x00083000 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	1:0	0x0	RW	Reserved.
NUMTC	5:2	0x1	RW	Number of Traffic Classes (TCs) for the port. This field must be set consistently with the settings made in the Tx-scheduler.
Reserved	15:10	0x0	Reserved	Reserved.
PFCLDA	31:16	0x079D	RW	PFC Link Delay Allowance. It is expressed in 16-byte units. Default value assumes 9.5 KB jumbo frames over a 10 GbE link with a 10GBASE-T PHY and 100 meter CAT6 cable (no optimization done for lower links speeds).

**10.2.1.12.4 DCB TC to PFC Mapping - PRTDCB_TC2PFC[PRT] (0x001C0980 + 0x4*PRT, PRT=0...3)**

Reset Source: CORER.

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TC2PFC	7:0	0x0	RW	Bitmap that controls the use of PFC per each TC. Bit n set to 1b =TC n uses PFC in Rx and Tx. The TC is referred as a no-drop TC. Bit n clear to 0b = The device does not issue PFC pause frames with bits set to 1b in the priority_enable_vector for the UPs attached to that TC. It does not react to bits set to 1b for the UPs attached to that TC in the priority_enable_vector of a received PFC pause frame. The TC is referred as a drop UP.
Reserved	31:8	0x0	Reserved	Reserved.

10.2.1.12.5 DCB Receive UP to TC Mapping for RCB - PRTDCB_RUP2TC[PRT] (0x001C09A0 + 0x4*PRT, PRT=0...3)

Bitmap long by 24-bits, with a 3-bit entry per each UP. Each entry controls the mapping of a UP to a 3-bit TC index in receive. A higher TC index means a higher priority of the TC.

The same mapping is used for packets received from the wires and for those looped back internally. It defines on account of which TC a packet is stored in the Rx packet buffer, and which UP's bits are set in the PFC XOFF/XON frames issued to the link partner when the filling state of a TC requires it.

Default mapping maps all UPs to TC0.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
UP0TC	2:0	0x0	RW	TC index to which UP 0 is mapped.
UP1TC	5:3	0x0	RW	TC index to which UP 1 is mapped.
UP2TC	8:6	0x0	RW	TC index to which UP 2 is mapped.
UP3TC	11:9	0x0	RW	TC index to which UP 3 is mapped.
UP4TC	14:12	0x0	RW	TC index to which UP 4 is mapped.
UP5TC	17:15	0x0	RW	TC index to which UP 5 is mapped.
UP6TC	20:18	0x0	RW	TC index to which UP 6 is mapped.
UP7TC	23:21	0x0	RW	TC index to which UP 7 is mapped.
Reserved	31:24	0x0	Reserved	Reserved.



**10.2.1.12.6 DCB Transmit Command Pipe Monitor Control - PRTDCB_TCPMC[PRT]
(0x000A21A0 + 0x4*PRT, PRT=0...3)**

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
CPM	12:0	0x098	RW	Depth of the per Port Monitor applied over the Tx LAN/RDMA Command Pipes altogether. It is expressed in commands units.
LLTC	20:13	0x0	RW	when set, TC is LL, when clear TC is bulk
RESERVED2	29:21	0x0	Reserved	reserved
RESERVED1	31	0b	Reserved	Reserved.

**10.2.1.12.7 DCB Transmit Command Monitoring Status per TC - PRTDCB_TCWSTC[n,PRT]
(0x000A2040 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)**

One register per TC. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MSTC	19:0	0x0	RO	Monitoring Status of the TC. Number of commands that are in transit from the host to the TCB (TCB waiting list included) for TC n, where n is the index of the register in the array.
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.12.8 DCB Transmit Data Pipe Monitor Control - PRTDCB_TDPMC[PRT] (0x000A0180 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	29:8	0x0	Reserved	Reserved.
Reserved	31	0b	Reserved	Reserved.

**10.2.1.12.9 DCB Transmit ETS Control for TCB - PRTDCB_TETSC_TCB[PRT] (0x000AE060 + 0x4*PRT, PRT=0...3)**

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	7:1	0x0	Reserved	Reserved
LLTC	15:8	0x0	RW	Bitmap long by 8-bits, with a 1-bit entry per each TC. Each entry controls whether or not the TC is considered to have low latency needs for the transmit path. 1b = TC is defined to be a low latency TC for transmit. 0b = TC is defined to be a bulk TC for transmit.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.12.10 DCB Transmit ETS Control for TPB - PRTDCB_TETSC_TPB[PRT] (0x00098060 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	7:1	0x0	Reserved	Reserved.
LLTC	15:8	0x0	RW	Bitmap long by 8-bits, with a 1-bit entry per each TC. Each entry controls whether or not the TC is considered to have low latency needs for the transmit path. 1b = TC is defined to be a low latency TC for transmit. 0b = TC is defined to be a bulk TC for transmit.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.12.11 DCB Transmit Frame Monitoring Status per TC - PRTDCB_TCMSTC[n,PRT] (0x000A0040 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per TC. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MSTC	19:0	0x0	RO	Monitoring Status of the TC. Number of bytes in transit from the host to the TPB (TPB waiting list included) for TC n, where n is the index of the register in the array.
Reserved	31:20	0x0	Reserved	Reserved.



10.2.1.12.12 DCB Transmit PFC Timer Status - PRTDCB_TPFACTS[n,PRT] (0x001E4660 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per TC. Register index corresponds to TCID.

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
PFCTIMER	13:0	0x0	RW	Current value of the PFC Timer of TC n in Tx, where n is the register index in the array. The amount is expressed in milliseconds. The timer must saturate to 0x3FFF. Writing to this field has the effect of loading a new current timer value, which can be useful for diagnostic purposes.
Reserved	31:14	0x0	Reserved	Reserved.

10.2.1.12.13 Transmit Flow Control Status - PRTDCB_TFCS[PRT] (0x001E4560 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TXOFF	0	0b	RO	Transmission Paused. Pause state indication of the transmit function when symmetrical link flow control is enabled.
Reserved	7:1	0x0	Reserved	Reserved.
TXOFF0	8	0b	RO	TC 0 Transmission Paused. Pause state indication of the TC 0 when PFC is enabled.
TXOFF1	9	0b	RO	TC 1 Transmission Paused. Pause state indication of the TC 1 when PFC is enabled.
TXOFF2	10	0b	RO	TC 2 Transmission Paused. Pause state indication of the TC 2 when PFC is enabled.
TXOFF3	11	0b	RO	TC 3 Transmission Paused. Pause state indication of the TC 3 when PFC is enabled.
TXOFF4	12	0b	RO	TC 4 Transmission Paused. Pause state indication of the TC 4 when PFC is enabled.
TXOFF5	13	0b	RO	TC 5 Transmission Paused. Pause state indication of the TC 5 when PFC is enabled.
TXOFF6	14	0b	RO	TC 6 Transmission Paused. Pause state indication of the TC 6 when PFC is enabled.
TXOFF7	15	0b	RO	TC 7 Transmission Paused. Pause state indication of the TC 7 when PFC is enabled.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.12.14 MAC Flow Control Register - PRTDCB_MFLCN[PRT] (0x001E2400 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
PMCF	0	0b	RW	Pass MAC Control Frames. Filter out unrecognized pause (flow control opcode doesn't match) and other control frames. 0b = Filter unrecognized pause frames. 1b = Pass/forward unrecognized pause frames.
DPF	1	0b	RW	Discard Pause Frame. When set to 0b, pause frames are sent to the host. When set to 1b, pause frames are discarded when <i>RFCE</i> or <i>RPFCM</i> is set to 1b. Setting this bit to 1b has no effect otherwise (if both <i>RFCE</i> and <i>RPFCM</i> are set to 0b).
RPFCM	2	0b	RW	Receive Priority Flow Control Mode. Indicates that the device responds to receiving PFC packets. If auto-negotiation is enabled, this bit should be set by software to the negotiated flow control value. This bit shall be set as a logical OR over the <i>RPFCE</i> [7:0] bitmap. It is useful to control forwarding of PFC frames to host if required. Note: Receive Priority Flow Control and Receive Link Flow Control are mutually exclusive and user should not configure both of them to be enabled at the same time. Note: This bit should not be set if bit 3 is set.
RFCE	3	0b	RW	Receive Link Flow Control Enable. Indicates that the device responds to receiving link flow control packets. If auto-negotiation is enabled, this bit should be set by software to the negotiated flow control value. Note: This bit should not be set if bit 2 is set.
RPFCE	11:4	0x0	RW	Receive Priority Flow Control Enable Bitmap. When bit n is set and after receiving PFC packets for UPn, the device stops transmitting over the TC to which UPn is mapped in Tx. When bit n is cleared, PFC indications received for UPn are ignored.
Reserved	31:12	0x0	Reserved	Reserved.



10.2.1.12.15 Flow Control Configuration - PRTDCB_FCCFG[PRT] (0x001E4640 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	2:0	0x0	Reserved	Reserved.
TFCE	4:3	0x0	RW	Transmit Flow Control Enable. These bits indicate that the X710-TM4 transmits flow control packets (XON/XOFF frames) based on receive fullness. If auto-negotiation is enabled, this bit should be set by software to the negotiated flow control value. 00b = Transmit flow control disabled. 01b = Link flow control enabled. 10b = PFC enabled. 11b = Reserved.
Reserved	31:5	0x0	Reserved	Reserved.

10.2.1.12.16 Flow Control Refresh Threshold Value - PRTDCB_FCRTV[PRT] (0x001E4600 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
FC_REFRESH_TH	15:0	0x7FFF	RW	Flow Control Refresh Threshold. This value is used to calculate the actual refresh period for sending the next pause frame if conditions for a pause state are still valid (buffer fullness above low threshold value). The formula for the refresh period for user priority N is $FCTTV[N/2].TTV[Nmod2] - FCRTV.FC_REFRESH_TH$.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.12.17 Flow Control Transmit Timer Value n - PRTDCB_FCTTVN[n,PRT] (0x001E4580 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TTV_2N	15:0	0xFFFF	RW	Transmit Timer Value 2n Timer value included in XOFF frames as Timer (2n). The same value shall be set to User Priorities attached to the same TC, as defined in PRTDCB_RUP2TC register. For legacy 802.3x flow control packets, TTV0 is the only timer that is used.



Field	Bit(s)	Init.	Access Type	Description
TTV_2N_P1	31:16	0xFFFF	RW	Transmit timer value 2n+1 timer value included in XOFF frames as timer (2n+1). The same value must be set to UPs attached to the same TC, as defined in PRTDCB_RUP2TC register.

10.2.1.12.18 DCB Receive ETS Control - PRTDCB_RETSC[PRT] (0x001223E0 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
NON_ETS_MODE	1	0b	RW	Rx Non-ETS Operating Mode 0b = Strict Priority (SP) mode. 1b = Round Robin (RR) mode.
Reserved	7:6	0x0	Reserved	Reserved.
LLTC	15:8	0x0	RW	Bitmap long by 8-bits, with a 1-bit entry per each TC. Each entry controls whether or not the TC is considered to have low latency needs for the receive path. 1b = TC is defined to be a low latency TC for receive. 0b = TC is defined to be a bulk TC for receive.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.12.19 DCB Receive ETS per TC Control - PRTDCB_RETSTCC[n,PRT] (0x00122180 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per TC. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
Reserved	29:7	0x0	Reserved	Reserved.
UPINTC_MODE	30	0b	RW	Rx-ETS operating mode when several UPs are attached to TC n, where n is the register index in the array. 0b = Strict Priority (SP) mode. 1b = Round Robin (RR) mode.
ETSTC	31	0b	RW	Controls the use of ETS as the Transmit Selection Algorithm (TSA) in Rx for TC n, where n is the register index in the array. 1b = TC n uses ETS scheme in Rx. 0b = TC n uses a Strict Priority or other TSA in Rx.



10.2.1.12.20 DCB Receive per Port Pipe Monitor Control - PRTDCB_RPPMC[PRT] (0x001223A0 + 0x4*PRT, PRT=0...3)

Meaningless when DCBRSPMC.RPM_MODE is not set to 00b.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
RX_FIFO_SIZE	23:16	0x08	RW	Rx Command FIFO Size. The number of Rx commands per TC that can be accumulated in RCB before sending back pressure to the RCU.
Reserved	31:24	0x0	Reserved	Reserved.

10.2.1.12.21 DCB Receive UP in PPRS - PRTDCB_RUP[PRT] (0x001C0B00 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
NOVLANUP	2:0	0x0	RW	This field assigns a default UP value to untagged incoming packets. The default UP is not inserted in the packet itself, but it controls in which linked list of RPB untagged packets are stored. UP 0 is the default.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.12.22 DCB Receive per UP PFC Timer Indication - GLDCB_RUPTI (0x00122618)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PFC TIMEOUT_UP	31:0	0x0	RW1C	PFC Time-out UPs. Bitmap where a bit set to 1b means that the PFC timer corresponding to the port/UP has timed out. Bits 0-7 are for port 0, UP 0 to 7. Bits 8-15 are for port 1, UP 0 to 7. Bits 16-23 are for port 2, UP 0 to 7. Bits 24-31 are for port 3, UP 0 to 7. Writing a bit with 1b restarts the corresponding PFC timer.



10.2.1.12.23 DCB Receive per UP PFC Timer Queue - PRTDCB_RUPTQ[n,PRT] (0x00122400 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per UP. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
RXQNUM	13:0	0x0	RO	Rx Queue Number. This register returns the Rx queue number (without enough available Rx descriptors) that caused the PFCTIMER of the port/UP to time out. The queue index reported in this register is the absolute queue index in the device space, which is different than the queue index used for the Tx and Rx queue registers. The value is meaningful only when the corresponding bit in PFDCB_RUPTI is set.
Reserved	31:14	0x0	Reserved	Reserved.

10.2.1.13 Receive Packet Buffer Registers

10.2.1.13.1 RPB Global High Watermark - GLRPB_GHW (0x000AC830)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GHW	19:0	0xF2000	RW	Global High Watermark (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.2 RPB Global Low Watermark - GLRPB_GLW (0x000AC834)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GLW	19:0	0x0	RW	Global Low Watermark (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.



10.2.1.13.3 RPB Packet High Watermark - GLRPB_PHW (0x000AC844)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PHW	19:0	0x1246	RW	Packet High Watermark. It is relative to the total number of packets stored in the RPB.
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.4 RPB Packet Low Watermark - GLRPB_PLW (0x000AC848)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PLW	19:0	0x0846	RW	Packet Low Watermark. It is relative to the total number of packets stored in the RPB.
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.5 RPB Dedicated Pool Size for Shared Buffer State - GLRPB_DPSS (0x000AC828)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
DPS_TCN	19:0	0x0	RW	Dedicated pool size for the single shared buffer state for all TCs of all ports (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.6 RPB Dedicated Pool High Watermark - PRTRPB_DHW[n,PRT] (0x000AC100 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per TC. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
DHW_TCN	19:0	0x0	RW	Dedicated Pool High Watermark for TC n (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.



10.2.1.13.7 RPB Dedicated Pool Low Watermark - PRTRPB_DLW[n,PRT] (0x000AC220 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per TC. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
DLW_TCN	19:0	0x0	RW	Dedicated Pool Low Watermark for TC n (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.8 RPB Dedicated Pool Size - PRTRPB_DPS[n,PRT] (0x000AC320 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per TC. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
DPS_TCN	19:0	0x0	RW	Dedicated Pool Size for TC n (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.9 RPB Shared Pool High Threshold - PRTRPB_SHT[n,PRT] (0x000AC480 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per TC. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
SHT_TCN	19:0	0x3C800	RW	Shared Pool High Threshold for TC n (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.10 RPB Shared Pool Low Threshold - PRTRPB_SLT[n,PRT] (0x000AC5A0 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

One register per TC. Register index corresponds to TCID.

Reset Source: CORER

NVM Load: No

Scope: PRT



Field	Bit(s)	Init.	Access Type	Description
SLT_TCN	19:0	0x0	RW	Shared Pool Low Threshold for TC n (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.11 RPB Shared Pool Size - PRTRPB_SPS[PRT] (0x000AC7C0 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
SPS	19:0	0x3C800	RW	Shared Pool Size. Number of bytes allocated to the shared pool of the port.
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.13.12 RPB Shared Pool High Watermark - PRTRPB_SHW[PRT] (0x000AC580 + 0x4*PRT, PRT=0...3)

Reset Source: CORER.

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
SHW	19:0	0x3C800	RW	Shared Pool High Watermark (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.

**10.2.1.13.13RPB Shared Pool Low Watermark - PRTRPB_SLW[PRT] (0x000AC6A0 + 0x4*PRT, PRT=0...3)**

Reset Source: CORER.

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
SLW	19:0	0x0	RW	Shared Pool Low Watermark (expressed in bytes).
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.14 Host Memory Cache Registers

Registers for host memory cache.

10.2.1.14.1 Private Memory LAN TX Object Size - GLHMC_LANTXOBSZ (0x000C2004)

This register is used to calculate the amount of memory that is required to be allocated for HMC LAN Tx queue objects. The value in this register is decoded such that the value = $\log_2(\text{ObjSize})$.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PMLANTX OBSZ	3:0	0x7	RO	0x7 = 128 bytes.
Reserved	31:4	0x0	Reserved	Reserved.

10.2.1.14.2 Private Memory LAN Queue Maximum - GLHMC_LANQMAX (0x000C2008)

This register reports the maximum number of LAN queue pairs that any PCI function can use. The actual number of queues that a software device driver can use might be limited by NVM settings.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PMLANQMAX	10:0	0x600	RO	Private Memory LAN Queue Maximum. This field reports the maximum number of LAN transmit or receive queue resources supported by the HMC. A given PCI function might be restricted to a smaller value by the NVRAM settings and the total number of enabled PCI functions.
Reserved	31:11	0x0	Reserved	Reserved.



10.2.1.14.3 Private Memory LAN RX Object Size - GLHMC_LANRXOBSZ (0x000C200C)

This register is used to calculate the amount of memory that is required to be allocated for HMC LAN Rx queue objects. The value in this register is decoded such that the value = $\log_2(\text{ObjSize})$.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PMLANRX OBSZ	3:0	0x5	RO	0x5 = 32 bytes.
Reserved	31:4	0x0	Reserved	Reserved.

10.2.1.14.4 Private Memory Segment Table Partitioning Registers - GLHMC_SDPART[n] (0x000C0800 + 0x4*n, n=0...15)

This register is used to partition the shared HMC segment table.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PMSDBASE	11:0	0x0	RW	Base segment table index for function n.
Reserved	15:12	0x0	Reserved	Reserved.
PMSDSIZE	28:16	0x0	RW	Number of valid segment table entries for function n.
Reserved	31:29	0x0	Reserved	Reserved.

Note: This register must be read only in the PF CSR space.

10.2.1.14.5 Private Memory Space Segment Descriptor Command - PFHMC_SDCMD[PF] (0x000C0000 + 0x4*PF, PF=0...15)

This register is used to access the HMC's segment table. The HMC's segment table is partitioned per PMF. This register is only allowed to access the segment table entries allocated to the PMF associated with this register via the SD Partition register and GLHMC_SDPART.

For read operations, PFHMC_SDCMD must be written with PMSDWR set to zero and PMSDIDX set to the segment table index to be read. After the write to PFHMC_SDCMD completes, PFHMC_SDDATALOW and PFHMC_SDDATAHIGH might be read to retrieve the segment descriptor contents.

For write operations, PFHMC_SDDATALOW and PFHMC_SDDATAHIGH must be written before writing PFHMC_SDCMD with PMSDWR set to one and PMSDIDX set to the segment table index to be written.

Reset Source: PFR.

NVM Load: No

Scope: PF



Field	Bit(s)	Init.	Access Type	Description
PMSDIDX	11:0	0x0	RW	Relative Index of the HMC Segment Descriptor to be read or written. The actual index to be used to access the Segment Table is (PMSDBASE + PMSDIDX), where PMSDBASE is from the GLHMC_SDPARTITION register associated with this function. On write operations, if (PMSDIDX >= GLHMC_SDPART.PMSDSIZE), the write will be dropped.
Reserved	30:12	0x0	Reserved	Reserved
PMSDWR	31	0b	RW	Set to 1 for write operations; set to 0 for read operations.

Note: 16 instances of this register are implemented for the X710-TM4. The remaining instances are reserved for future expansion.

10.2.1.14.6 Private Memory Space Segment Descriptor Data Low - PFHMC_SDDATALOW[PF] (0x000C0100 + 0x4*PF, PF=0...15)

This register is used in conjunction with PFHMC_SDCMD and PFHMC_SDDATAHIGH to access the HMC's segment table.

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PMSDVALID	0	0b	RW	Valid bit of an HMC segment descriptor table entry.
PMSDTYPE	1	0b	RW	Segment Descriptor Type. When set to 0b, the SD is paged (the SD points to a the physical address of a host memory page that contains an array of page descriptors). When set to 1b, the SD directly points the physical address of a physically contiguous 2 MB memory region.
PMSDBP COUNT	11:2	0x0	RW	Backing page count of an HMC segment descriptor table entry. Every SD entry in a given FPM space must be set to 512 except the last SD. The last SD can have a value from 1 to 512. This field is used to calculate the end of the FPM space associated with a SD without having to read the valid bit for each individual PD entry.
PMSDDATA LOW	31:12	0x0	RW	Bits 31:12 bits of an HMC segment descriptor table entry.

Notes: 16 instances of this register are implemented for the X710-TM4. The remaining instances are reserved for future expansion.



10.2.1.14.7 Private Memory Space Segment Descriptor Data High - PFHMC_SDDATAHIGH[PF] (0x000C0200 + 0x4*PF, PF=0...15)

This register is used in conjunction with PFHMC_SDCMD and PFHMC_SDDATALOW to access the HMC's segment table.

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PMSDDATA HIGH	31:0	0x0	RW	Most significant 32 bits of a segment descriptor.

Note: 16 instances of this register are implemented for the X710-TM4. The remaining instances are reserved for future expansion.

10.2.1.14.8 Private Memory Space Page Descriptor Invalidate - PFHMC_PDINV[PF] (0x000C0300 + 0x4*PF, PF=0...15)

This register is used to invalidate cached HMC page descriptors that are set to the invalid state by software.

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PMSDIDX	11:0	0x0	RW	Relative Index of the HMC segment descriptor associated with the HMC PD that is to be invalidated. For PFs, the actual index to be used to access the segment table is (PMSDBASE + PMSDIDX), where PMSDBASE is from the GLHMC_SDPART register associated with this function. On write operations, if (PMSDIDX >= GLHMC_SDPART.PMSDSIZE), the write is dropped.
Reserved	15:12	0x0	Reserved	Reserved.
PMPDIDX	24:16	0x0	RW	Index of the PD within the PD page indicated by PMSDIDX.
Reserved	31:25	0x0	Reserved	Reserved.

Note: 16 instances of this register are implemented for the X710-TM4. The remaining instances are reserved for future expansion.

10.2.1.14.9 Host Memory Cache Error Information Register - PFHMC_ERRORINFO[PF] (0x000C0400 + 0x4*PF, PF=0...15)

This register reports the errors detected by the HMC. Errors reported through this register also might trigger interrupts through the HMC_ERR bit in the PFINT_ICR0 register.

Reset Source: PFR

NVM Load: No

Scope: PF



Field	Bit(s)	Init.	Access Type	Description
PMF_INDEX	4:0	0x0	RW	Private Memory Function Index. This field reports the HMC PMF associated with the error. Writes to this field are ignored.
Reserved	6:5	0x0	Reserved	Reserved.
PMF_ISVF	7	0b	RW	Private Memory Function Is VF. When set, the PMF reported in the PMF_INDEX is associated with a PE-enabled VF. When cleared, the PMF reported in the PMF_INDEX is associated with a PF. Writes to this field are ignored.
HMC_ERROR_TYPE	11:8	0x0	RW	HMC Error Type. This field reports the error type detected by the HMC. The values are: 0 = PMF is not valid (the valid bit is cleared in the GLHMC_VFPMFMAP register associated with the PMF). 1 = Invalid PMF index for a PE-enabled VF in GLHMC_VFPMFMAP (such as, The PMF index programmed in the GLHMC_VFPMFMAP register is < 16 or > 47. This is a firmware error.) 2 = Invalid PF for a PE enabled VF in GLHMC_VFPMFTABLE (like, the parent PF index programmed in the GLHMC_VFPMFMAP register did not match the PF index received in the HMC transaction. This is a firmware error). 3 = Invalid LAN queue index (such as, the absolute LAN queue index received in the HMC transaction was less than the LAN queue index base register associated with the PF index received in the HMC transaction). 4 = Object index from transaction was larger than the value specified in the object's GLHMC_*CNT register. 5 = Private memory address extends beyond the limits of the SDs assigned to the PCIe function. 6 = Segment Descriptor Invalid. 7 = Segment Descriptor Too Small (only applies to direct-mapped SDs) 8 = Page Descriptor Invalid. 9 = Received Unsupported Request (UR) completion from PCIe read of object. 10 = Valid bit in PFLAN_QALLOC_PMAT[PF] or PF_VT_PFALLOC_PMAT[PF] register was not set. 11 = An invalid object type was detected. The PFHMC_ERRORDATA register can be read to determine the LAN queue index associated with error type 3. The PFHMC_ERRORDATA register can be read to determine the HMC object index associated with error types 4 and 12. The PFHMC_ERRORDATA register can be read to determine the HMC function relative SD_Index and PD_Index associated with error types 5 through 9.
Reserved	15:12	0x0	Reserved	Reserved.
HMC_OBJECT_TYPE	20:16	0x0	RW	Specifies the object type associated with the error.
Reserved	30:21	0x0	Reserved	Reserved.
ERROR_DETECTED	31	0b	RW	Error Detected. This field is set to 1b when a new error has been detected by the HMC. No subsequent errors are recorded until this field is written with a value of 0b. Writes of 0b to this register clears the error and enables a subsequent error to be reported. Writes of 1b to this field are ignored.



**10.2.1.14.10 Host Memory Cache Error Data Register - PFHMC_ERRORDATA[PF]
(0x000C0500 + 0x4*PF, PF=0...15)**

This register reports the HMC function relative SD_Index and PD_Index or HMC object index related to an error detected by the HMC.

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
HMC_ERROR_DATA	29:0	0x0	RO	<p>Error Data. This field reports either the HMC function relative SD_Index, PD_Index, LAN queue index or HMC object index associated with the error reported in the PFHMC_ERRORINFO register.</p> <p>When PFHMC_ERRORINFO.HMC_ERROR_TYPE is 3, HMC_ERROR_DATA[27:0] reports the LAN queue index associated with the error and HMC_ERROR_DATA[29:28] should be zero.</p> <p>When PFHMC_ERRORINFO.HMC_ERROR_TYPE is 4 or 12, HMC_ERROR_DATA[27:0] reports the object index associated with the error and HMC_ERROR_DATA[29:28] should be zero.</p> <p>When PFHMC_ERRORINFO.HMC_ERROR_TYPE is 5 through 9, HMC_ERROR_DATA[29:9] reports the HMC function relative SD_Index and PD_Index associated with the error detected by the HMC. HMC_ERROR_DATA[29:18] is set to the HMC function relative SD_Index for the affected HMC function. HMC_ERROR_DATA[17:9] is set to the PD_Index. HMC_ERROR_DATA[8:0] are reserved for error types 5 through 9.</p> <p>When PFHMC_ERRORINFO.HMC_ERROR_TYPE is 0 through 2 or 10 through 11, HMC_ERROR_DATA[29:0] is not valid and should be zero.</p>
Reserved	31:30	0x0	Reserved	Reserved.

10.2.1.14.11 FPM LAN TX Queue Base - GLHMC_LANTXBASE[n] (0x000C6200 + 0x4*n, n=0...15)

This register reports the FPM space base address for the LAN transmit queue objects in 512-byte increments. The value in this register must be multiplied by 512 to get the actual address out of the 8 GB FPM address space.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FPMLANTX BASE	23:0	0x0	RW	
Reserved	31:24	0x0	RW	Reserved.

Note: 16 instances of this register are implemented for the X710-TM4. The remaining instances are reserved for future expansion.



10.2.1.14.12 FPM LAN TX Queue Object Count - GLHMC_LANTXCNT[n] (0x000C6300 + 0x4*n, n=0...15)

This register is used to set the FPM space size for the LAN Tx queue objects.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FPMLANTX CNT	10:0	0x0	RW	
Reserved	31:11	0x0	Reserved	Reserved.

Note: 16 instances of this register are implemented for the X710-TM4. The remaining instances are reserved for future expansion.

10.2.1.14.13 FPM LAN RX Queue Base - GLHMC_LANRXBASE[n] (0x000C6400 + 0x4*n, n=0...15)

This register reports the FPM space base address for the LAN receive queue objects in 512-byte increments. The value in this register must be multiplied by 512 to get the actual address out of the 8 GB FPM address space.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FPMLANRX BASE	23:0	0x0	RW	
Reserved	31:24	0x0	Reserved	Reserved.

Note: 16 instances of this register are implemented for the X710-TM4. The remaining instances are reserved for future expansion.

10.2.1.14.14 FPM LAN RX Queue Object Count - GLHMC_LANRXCNT[n] (0x000C6500 + 0x4*n, n=0...15)

This register is used to set the FPM space size for the LAN Rx queue objects.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FPMLANRX CNT	10:0	0x0	RW	
Reserved	31:11	0x0	Reserved	Reserved.



Note: 16 instances of this register are implemented for the X710-TM4. The remaining instances are reserved for future expansion.

10.2.1.15 Context Manager Registers

10.2.1.15.1 CMLAN Error Data - PFCM_LAN_ERRDATA[PF] (0x0010C080 + 0x4*PF, PF=0...15)

This register reports errors for internal CM clients.

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ERROR_CODE	3:0	0x0	RO	Error Code. 0x1 = PMAT error - other. 0x2 = PMAT error - dummy completion. 0x3 = Context CRC error. All other values are reserved.
Q_TYPE	6:4	0x0	RO	
Reserved	7	0b	Reserved	Reserved.
Q_NUM	19:8	0x0	RO	
Reserved	31:20	0x0	Reserved	Reserved.

10.2.1.15.2 CMLAN Error Info - PFCM_LAN_ERRINFO[PF] (0x0010C000 + 0x4*PF, PF=0...15)

This register reports errors for internal cm clients.

Reset Source: PFR

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ERROR_VALID	0	0b	RO	Indicates the information in <i>ERRINFO</i> and <i>ERRDATA</i> is valid. Writing a 1b to this bit clears the contents of <i>ERRINFO</i> and <i>ERRDATA</i> . Writing a 0b to this bit has no affect.
Reserved	3:1	0x0	Reserved	Reserved.
ERROR_INST	6:4	0x0	RO	Indicates the internal unit that reported the error. 0x1 = DBL. 0x2 = RLU. 0x3 = RLS.
Reserved	7	0b	Reserved	Reserved.
DBL_ERROR_CNT	15:8	0x0	RO	Number of DBL errors reported.
RLU_ERROR_CNT	23:16	0x0	RO	Number of RLU errors reported.
RLS_ERROR_CNT	31:24	0x0	RO	Number of RLS errors reported.



10.2.1.15.3 CMLAN Context Control Register - PFCM_LANCTXCTL[PF] (0x0010C300 + 0x4*PF, PF=0...15)

This register provides an interface into the context cache for pre-boot context initialization.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
QUEUE_NUM	11:0	0x0	RW	Specifies the LAN queue index to be loaded or invalidated. This number is an absolute queue number.
SUB_LINE	14:12	0x0	RW	Specifies the 16-byte sub-line to be accessed for write operations.
QUEUE_TYPE	16:15	0x0	RW	Specifies the queue type. 0x0 = LAN Rx context. 0x1 = LAN Tx context. 0x2 = Reserved. 0x3 = Reserved.
OP_CODE	18:17	0x0	RW	Specifies the operation type. 0x0 = Read. 0x1 = Write. 0x2 = Invalidate. 0x3 = Reserved.
Reserved	31:19	0x0	Reserved	Reserved.

10.2.1.15.4 CMLAN Context Status Register - PFCM_LANCTXSTAT[PF] (0x0010C380 + 0x4*PF, PF=0...15)

This register provides completion status for operations initiated via the CMLAN context control register.

Reset Source: CORER

NVM Load: No

Scope: PF



Field	Bit(s)	Init.	Access Type	Description
CTX_DONE	0	0b	RO	Context Operation Done. This field is set to 0b when an a previous context LAN context operation has been initiated by a write to the PFCM_LANCTXCTL register is in progress or when no LAN context operation has been issued. This field is set to 1b when an previous LAN context operation that was initiated by a write to the PFCM_LANCTXCTL register completed.
CTX_MISS	1	0b	RO	Context Queue Number Not Present. This field reports if the queue was in the context cache before the operation specified in PFCM_LANCTXCTL.OP_CODE. This field is set to 0b if the requested queue number was resident in the context cache and to 1b if the requested queue number was not resident in the context cache.
Reserved	31:2	0x0	Reserved	Reserved.

10.2.1.15.5 CMLAN Context Data Registers - PFCM_LANCTXDATA[n,PF] (0x0010C100 + 0x80*n + 0x4*PF, n=0...3, PF=0...15)

32-bit portion of the 128-bit context sub-line data. This register is indexed by PF index (n) and by the context word index. Word index 0 is the LS word and line 3 is the MS word.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
DATA	31:0	0x0	RW	

10.2.1.16 Admin Queue

Registers related to the admin queue.

10.2.1.16.1 PF Admin Transmit Queue Base Address Low - PF_ATQBAL[PF] (0x00080000 + 0x4*PF, PF=0...15)

This register contains the lower bits of the 64-bit descriptor base address.

Reset Source: EMPR.

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ATQBAL	31:0	0x0	RW	Transmit descriptor base address low; must be 64-byte aligned.

**10.2.1.16.2 PF Admin Transmit Queue Base Address High - PF_ATQBAH[PF] (0x00080100 + 0x4*PF, PF=0...15)**

This register contains the higher bits of the 64-bit descriptor base address.

Reset Source: EMPR.

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ATQBAH	31:0	0x0	RW	Transmit descriptor base address high.

10.2.1.16.3 PF Admin Transmit Queue Length - PF_ATQLEN[PF] (0x00080200 + 0x4*PF, PF=0...15)

This register sets the size of the ring. Maximum size is 1024.

Reset Source: EMPR.

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ATQLEN	9:0	0x0	RW	Descriptor ring length; maximum size is 1024.
Reserved	27:10	0x0	Reserved	Reserved.
ATQVFE	28	0b	RW	VF Error. Set by firmware on a PF queue when one of its VFs had an admin queue error.
ATQOVFL	29	0b	RW	Overflow Error. Set by firmware when a message was lost because there was no room on the queue.
ATQCRIT	30	0b	RW	Critical Error. This bit is set by firmware when a critical error has been detected on this queue.
ATQENABLE	31	0b	RW	Enable Bit. Set by the software device driver to indicate that the queue is active. When setting the enable bit, software should initialize all other fields. This flag is implemented by a FF and cleared by PFR.

10.2.1.16.4 PF Admin Transmit Head - PF_ATQH[PF] (0x00080300 + 0x4*PF, PF=0...15)

Admin transmit queue head pointer.

Reset Source: EMPR.

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ATQH	9:0	0x0	RW	Transmit queue head pointer. At queue initialization, software clears the head pointer and during normal operation firmware increments the head following command execution.
Reserved	31:10	0x0	Reserved	Reserved.



10.2.1.16.5 PF Admin Transmit Tail - PF_ATQT[PF] (0x00080400 + 0x4*PF, PF=0...15)

Transmit queue Tail pointer.

Reset Source: EMPR.

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ATQT	9:0	0x0	RW	Transmit Queue Tail. Incremented to indicate that there are new valid descriptors on the ring. Software might only write to this register once when both transmit and receive queues are properly initialized. Clears to zero at queue initialization.
Reserved	31:10	0x0	Reserved	Reserved.

10.2.1.16.6 PF Admin Receive Queue Base Address Low - PF_ARQBAL[PF] (0x00080080 + 0x4*PF, PF=0...15)

This register contains the lower bits of the 64-bit descriptor base address.

Reset Source: EMPR.

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ARQBAL	31:0	0x0	RW	Receive descriptor base address low; must be 64-byte aligned.

10.2.1.16.7 PF Admin Receive Queue Base Address High - PF_ARQBAH[PF] (0x00080180 + 0x4*PF, PF=0...15)

This register contains the higher bits of the 64-bit descriptor base address.

Reset Source: EMPR, reset by hardware: EMPR, reset by firmware: None.

NVM Load: No, Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ARQBAH	31:0	0x0	RW	Receive descriptor base address high.

10.2.1.16.8 PF Admin Receive Queue Length - PF_ARQLEN[PF] (0x00080280 + 0x4*PF, PF=0...15)

This register specifies the receive queue length; maximum size is 1024.

Reset Source: EMPR.

NVM Load: No

Scope: PF



Field	Bit(s)	Init.	Access Type	Description
ARQLEN	9:0	0x0	RW	Descriptor ring length; maximum size is 1024.
Reserved	27:10	0x0	Reserved	Reserved.
ARQVFE	28	0b	RW	VF Error. Set by firmware on a PF queue when one of its VFs had an admin queue error.
ARQOVFL	29	0b	RW	Overflow Error. Set by firmware when a message was lost because there was no room on the queue.
ARQCRIT	30	0b	RW	Critical Error. This bit is set by firmware when a critical error has been detected on this queue.
ARQENABLE	31	0b	RW	Enable Bit. Set by the software device driver to indicate that the queue is active. When setting the enable bit, software should initialize all other fields. This flag is implemented by a FF and cleared by PFR.

10.2.1.16.9 PF Admin Receive Queue Head - PF_ARQH[PF] (0x00080380 + 0x4*PF, PF=0...15)

Admin receive queue head pointer.

Reset Source: EMPR.

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ARQH	9:0	0x0	RW	Receive Queue Head Pointer. At queue initialization, software clears the head pointer and during normal operation firmware increments the head following command execution.
Reserved	31:10	0x0	Reserved	Reserved.

10.2.1.16.10 PF Admin Receive Queue Tail - PF_ARQT[PF] (0x00080480 + 0x4*PF, PF=0...15)

Admin receive tail pointer.

Reset Source: EMPR.

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
ARQT	9:0	0x0	RW	Receive Queue Tail. Incremented to indicate that there are new valid descriptors on the ring. Software might only write to this register once when the queue is fully configured. Clears to zero at queue initialization.
Reserved	31:10	0x0	Reserved	Reserved.



10.2.1.16.11 VF Admin Transmit Queue Base Address Low - VF_ATQBAL[VF] (0x00080800 + 0x4*VF, VF=0...127)

This register contains the lower bits of the 64-bit descriptor base address.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ATQBAL	31:0	0x0	RW	Transmit descriptor base address low; must be 64-byte aligned.

10.2.1.16.12 VF Admin Transmit Queue Base Address High - VF_ATQBAH[VF] (0x00081000 + 0x4*VF, VF=0...127)

This register contains the higher bits of the 64-bit descriptor base address.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ATQBAH	31:0	0x0	RW	Transmit descriptor base address high.

10.2.1.16.13 VF Admin Transmit Queue Length - VF_ATQLEN[VF] (0x00081800 + 0x4*VF, VF=0...127)

This register sets the size of the ring. Maximum size is 1024.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ATQLEN	9:0	0x0	RW	Descriptor ring length; maximum size is 1024.
Reserved	27:10	0x0	Reserved	Reserved.
ATQVFE	28	0b	RW	VF Error. Set by firmware on a PF queue when one of its VFs had an admin queue error.
ATQOVFL	29	0b	RW	Overflow Error. Set by firmware when a message was lost because there was no room on the queue.
ATQCRIT	30	0b	RW	Critical Error. This bit is set by firmware when a critical error has been detected on this queue.
ATQENABLE	31	0b	RW	Enable Bit. Set by the software device driver to indicate that the queue is active. When setting the enable bit, software should initialize all other fields. This flag is implemented by a FF and cleared by VFR.



10.2.1.16.14 VF Admin Transmit Head - VF_ATQH[VF] (0x00082000 + 0x4*VF, VF=0...127)

Admin transmit queue head pointer.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ATQH	9:0	0x0	RW	Transmit Queue Head Pointer. At queue initialization, software clears the head pointer and during normal operation, firmware increments the head following command execution.
Reserved	31:10	0x0	Reserved	Reserved.

10.2.1.16.15 VF Admin Transmit Tail - VF_ATQT[VF] (0x00082800 + 0x4*VF, VF=0...127)

Transmit queue tail pointer.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ATQT	9:0	0x0	RW	Transmit Queue Tail. Incremented to indicate that there are new valid descriptors on the ring. Software might only write to this register once when both transmit and receive queues are properly initialized. Clears to zero at queue initialization.
Reserved	31:10	0x0	Reserved	Reserved.



10.2.1.16.16 VF Admin Receive Queue Base Address Low - VF_ARQBAL[VF] (0x00080C00 + 0x4*VF, VF=0...127)

This register contains the lower bits of the 64-bit descriptor base address.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ARQBAL	31:0	0x0	RW	Receive descriptor base address low; must be 64-byte aligned.

10.2.1.16.17 VF Admin Receive Queue Base Address High - VF_ARQBAH[VF] (0x00081400 + 0x4*VF, VF=0...127)

This register contains the higher bits of the 64-bit descriptor base address.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ARQBAH	31:0	0x0	RW	Receive descriptor base address high.

10.2.1.16.18 VF Admin Receive Queue Length - VF_ARQLEN[VF] (0x00081C00 + 0x4*VF, VF=0...127)

This register specifies the receive queue length; maximum size is 1024.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ARQLEN	9:0	0x0	RW	Descriptor ring length; maximum size is 1024.
Reserved	27:10	0x0	Reserved	Reserved.
ARQVFE	28	0b	RW	VF Error. Set by firmware on a PF queue when one of its VFs had an admin queue error.
ARQOVFL	29	0b	RW	Overflow Error. Set by firmware when a message was lost because there was no room on the queue.
ARQCRIT	30	0b	RW	Critical Error. This bit is set by firmware when a critical error has been detected on this queue
ARQENABLE	31	0b	RW	Enable Bit. Set by the software device driver to indicate that the queue is active. When setting the enable bit, software should initialize all other fields. This flag is implemented by a FF and cleared by PFR.



10.2.1.16.19VF Admin Receive Queue Head - VF_ARQH[VF] (0x00082400 + 0x4*VF, VF=0...127)

Admin receive queue head pointer.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ARQH	9:0	0x0	RW	Receive Queue Head Pointer. At queue initialization, software clears the head pointer and during normal operation firmware increments the head following command execution.
Reserved	31:10	0x0	Reserved	Reserved.

10.2.1.16.20VF Admin Receive Queue Tail - VF_ARQT[VF] (0x00082C00 + 0x4*VF, VF=0...127)

Admin receive tail pointer.

Reset Source: EMPR

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
ARQT	9:0	0x0	RW	Receive Queue Tail Incremented to indicate that there are new valid descriptors on the ring. Software might only write to this register once when the queue is fully configured. Clears to zero at queue initialization.
Reserved	31:10	0x0	Reserved	Reserved.

10.2.1.16.21Global Admin Transmit Queue Base Address Low - GL_ATQBAL (0x00080040)

This register contains the lower bits of the 64-bit descriptor base address.

Reset Source: EMPR, reset by hardware: EMPR, reset by firmware: None.

NVM Load: No, Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ATQBAL	31:0	0x0	RW	Transmit descriptor base address low; must be 64-byte aligned.

10.2.1.16.22Global Admin Transmit Queue Base Address High - GL_ATQBAH (0x00080140)

This register contains the higher bits of the 64-bit descriptor base address.

Reset Source: EMPR

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
ATQBAH	31:0	0x0	RW	Transmit descriptor base address high.

10.2.1.16.23 Global Admin Transmit Queue Length - GL_ATQLEN (0x00080240)

This register sets the size of the ring. Maximum size is 1024.

Reset Source: EMPR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ATQLEN	9:0	0x0	RW	Descriptor ring length; maximum size is 1024.
Reserved	27:10	0x0	Reserved	Reserved.
ATQVFE	28	0b	RW	VF Error. Set by firmware on a PF queue when one of its VFs had an admin queue error.
ATQOVFL	29	0b	RW	Overflow Error. Set by firmware when a message was lost because there was no room on the queue.
ATQCRIT	30	0b	RW	Critical Error. This bit is set by firmware when a critical error has been detected on this queue.
ATQENABLE	31	0b	RW	Enable Bit. Set by the software device driver to indicate that the queue is active. When setting the enable bit, software should initialize all other fields. This flag is implemented by a FF and cleared by CORER.

10.2.1.16.24 Global Admin Transmit Head - GL_ATQH (0x00080340)

Admin transmit queue head pointer.

Reset Source: EMPR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ATQH	9:0	0x0	RW	Transmit Queue Head Pointer. At queue initialization, software clears the head pointer and during normal operation firmware increments the head following command execution.
Reserved	31:10	0x0	Reserved	Reserved.

10.2.1.16.25 Global Admin Transmit Tail - GL_ATQT (0x00080440)

Transmit queue tail pointer.

Reset Source: EMPR

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
ATQT	9:0	0x0	RW	Transmit Queue Tail. Incremented to indicate that there are new valid descriptors on the ring. Software might only write to this register once when both transmit and receive queues are properly initialized. Clears to zero at queue initialization.
Reserved	31:10	0x0	Reserved	Reserved.

10.2.1.16.26 Global Admin Receive Queue Base Address Low - GL_ARQBAL (0x000800C0)

This register contains the lower bits of the 64-bit descriptor base address.

Reset Source: EMPR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Description
ARQBAL	31:0	0x0	Receive descriptor base address low; must be 64-byte aligned.

10.2.1.16.27 Global Admin Receive Queue Base Address High - GL_ARQBAH (0x000801C0)

This register contains the higher bits of the 64-bit descriptor base address.

Reset Source: EMPR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ARQBAH	31:0	0x0	RW	Receive descriptor base address high.

10.2.1.16.28 Global Admin Receive Queue Head - GL_ARQH (0x000803C0)

Admin receive queue head pointer.

Reset Source: EMPR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ARQH	9:0	0x0	RW	Receive Queue Head Pointer. At queue initialization, software clears the head pointer and during normal operation firmware increments the head following command execution.
Reserved	31:10	0x0	Reserved	Reserved.



10.2.1.16.29 Global Admin Receive Queue Tail - GL_ARQT (0x000804C0)

Admin receive tail pointer.

Reset Source: EMPR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ARQT	9:0	0x0	RW	Receive Queue Tail. Incremented to indicate that there are new valid descriptors on the ring. Software might only write to this register once when the queue is fully configured. Clears to zero at queue initialization.
Reserved	31:10	0x0	Reserved	Reserved.

10.2.1.17 Statistics Registers

Statistics counters. Refer to the Statistics section (under Shared Resources) for information regarding wide counters, clearing counters, sampling points and counter consistency rules.

Notes: Wide counters (48 bits) are represented as two registers high and low, containing the 16 MSBs and 32 LSBs, respectively. they are implemented as a 64-bit register. Atomicity is only guaranteed when reading both parts in one 64-bit read.

Per-function registers are implemented as an array of 144 registers. Elements 0-127 correspond to VFs 0-127 and elements 128-143 are used for PFs 0-15.

10.2.1.17.1 Port Good Octets Received Count Low - GLPRT_GORCL[n] (0x00300000 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GORCL	31:0	0x0	RW1C	Low 32 bits of the receive octet count by this port. The entire register should be read by a 64-bit access.

10.2.1.17.2 Port Good Octets Received Count High - GLPRT_GORCH[n] (0x00300004 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GORCH	15:0	0x0	RW1C	High 16 bits of the receive octet count by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.17.3 Port Unicast Packets Received Count Low - GLPRT_UPRCL[n] (0x003005A0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPRCL	31:0	0x0	RW1C	Low 32 bits of the receive unicast packet count by this port. The entire register should be read by a 64-bit accesses.

10.2.1.17.4 Port Unicast Packets Received Count High - GLPRT_UPRCH[n] (0x003005A4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPRCH	15:0	0x0	RW1C	High 16 bits of the receive unicast packet count by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.5 Port Multicast Packets Received Count Low - GLPRT_MPRCL[n] (0x003005C0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPRCL	31:0	0x0	RW1C	Low 32 bits of the receive multicast packet count by this port. The entire register should be read by a 64-bit access.

10.2.1.17.6 Port Multicast Packets Received Count High - GLPRT_MPRCH[n] (0x003005C4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPRCH	15:0	0x0	RW1C	High 16 bits of the receive multicast packet count by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.17.7 Port Broadcast Packets Received Count Low - GLPRT_BPRCL[n] (0x003005E0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPRCL	31:0	0x0	RW1C	Low 32 bits of the receive broadcast packet count by this port. The entire register should be read by a 64-bit access.

10.2.1.17.8 Port Broadcast Packets Received Count High - GLPRT_BPRCH[n] (0x003005E4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPRCH	15:0	0x0	RW1C	High 16 bits of the receive broadcast packet count by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	

10.2.1.17.9 Port Receive Packets Discarded Count - GLPRT_RDPC[n] (0x00300600 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RDPC	31:0	0x0	RW1C	Counts received packets from the network that are dropped in the receive packet buffer. The packets are dropped due to possible lack of bandwidth on PCIe or total bandwidth of the internal data path.

10.2.1.17.10 Port Received With No Destination - GLPRT_RUPP[n] (0x00300660 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RUPP	31:0	0x0	RW1C	Receive packets dropped for this port because they did not match any STAG (no forwarding rule). When VEB statistics are disabled, this counter should be ignored.



10.2.1.17.11 Port Good Octets Transmit Count Low - GLPRT_GOTCL[n] (0x00300680 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GOTCL	31:0	0x0	RW1C	Low 32 bits of the transmit octet count by this port. The entire register should be read by a 64-bit access.

10.2.1.17.12 Port Good Octets Transmit Count High - GLPRT_GOTCH[n] (0x00300684 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GOTCH	15:0	0x0	RW1C	High 16 bits of the transmit octet count by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.13 Port Unicast Packets Transmit Count Low - GLPRT_UPTCL[n] (0x003009C0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPTCL	31:0	0x0	RW1C	Low 32 bits of the transmit unicast packet by this port. The entire register should be read by a 64-bit access.

10.2.1.17.14 Port Unicast Packets Transmit Count High - GLPRT_UPTCH[n] (0x003009C4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPTCH	15:0	0x0	RW1C	High 16 bits of the transmit unicast packet by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.17.15 Port Multicast Packets Transmit Count Low - GLPRT_MPTCL[n] (0x003009E0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPTCL	31:0	0x0	RW1C	Low 32 bits of the transmit multicast packet count by this port. The entire register should be read by a 64-bit access.

10.2.1.17.16 Port Multicast Packets Transmit Count High - GLPRT_MPTCH[n] (0x003009E4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPTCH	15:0	0x0	RW1C	High 16 bits of the transmit multicast packet count by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.17 Port Broadcast Packets Transmit Count Low - GLPRT_BPTCL[n] (0x00300A00 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPTCL	31:0	0x0	RW1C	Low 32 bits of the transmit broadcast packet count by this port. The entire register should be read by a 64-bit access.

10.2.1.17.18 Port Broadcast Packets Transmit Count High - GLPRT_BPTCH[n] (0x00300A04 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPTCH	15:0	0x0	RW1C	High 16 bits of the transmit broadcast packet count by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.17.19 Transmit Discard On Link Down - GLPRT_TDOLD[n] (0x00300A20 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GLPRT_TDOLD	31:0	0x0	RW1C	Packets discarded at the port because the link was down.

10.2.1.17.20 Packets Received [64 Bytes] Count Low - GLPRT_PRC64L[n] (0x00300480 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC64L	31:0	0x0	RW1C	Low 32 bits of the good receive packet count that are 64 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.21 Packets Received [64 Bytes] Count High - GLPRT_PRC64H[n] (0x00300484 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC64H	15:0	0x0	RW1C	High 16 bits of the good receive packet count that are 64 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.22 Packets Received [65-127 Bytes] Count Low - GLPRT_PRC127L[n] (0x003004A0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC127L	31:0	0x0	RW1C	Low 32 bits of the good receive packet count that are 65-127 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.



10.2.1.17.23 Packets Received [65-127 Bytes] Count High - GLPRT_PRC127H[n] (0x003004A4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC127H	15:0	0x0	RW1C	High 16 bits of the good receive packet count that are 65-127 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	

10.2.1.17.24 Packets Received [128-255 Bytes] Count Low - GLPRT_PRC255L[n] (0x003004C0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC255L	31:0	0x0	RW1C	Low 32 bits of the good receive packet count that are 128-255 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.25 Packets Received [128-255 Bytes] Count High - GLPRT_PRC255H[n] (0x003004C4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRTPRC255H	15:0	0x0	RW1C	High 16 bits of the good receive packet count that are 128-255 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.26 Packets Received [256-511 Bytes] Count Low - GLPRT_PRC511L[n] (0x003004E0 + 0x8*n, n=0...3)

Reset Source: CORER, reset by hardware: CORER, reset by firmware: None.

NVM Load: No, Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC511L	31:0	0x0	RW1C	Low 32 bits of the good receive packet count that are 256-511 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.



**10.2.1.17.27 Packets Received [256-511 Bytes] Count High - GLPRT_PRC511H[n]
(0x003004E4 + 0x8*n, n=0...3)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC511H	15:0	0x0	RW1C	High 16 bits of the good receive packet count that are 256-511 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	

**10.2.1.17.28 Packets Received [512-1023 Bytes] Count Low - GLPRT_PRC1023L[n]
(0x00300500 + 0x8*n, n=0...3)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC1023L	31:0	0x0	RW1C	Low 32 bits of the good receive packet count that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

**10.2.1.17.29 Packets Received [512-1023 Bytes] Count High - GLPRT_PRC1023H[n]
(0x00300504 + 0x8*n, n=0...3)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC1023H	15:0	0x0	RW1C	High 16 bits of the good receive packet count that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.30 Packets Received [1024-1522] Count Low - GLPRT_PRC1522L[n] (0x00300520 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
PRC1522L	31:0	0x0	RW1C	Low 32 bits of the good receive packet count that are 1024-1522 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.31 Packets Received [1024-1522] Count High - GLPRT_PRC1522H[n] (0x00300524 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC1522H	15:0	0x0	RW1C	High 16 bits of the good receive packet count that are 1024-1522 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.32 Packets Received [1523-9522 Bytes] Count Low - GLPRT_PRC9522L[n] (0x00300540 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC1522L	31:0	0x0	RW1C	Low 32 bits of the good receive packet count that are 1523-max bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.33 Packets Received [1523-9522 Bytes] Count High - GLPRT_PRC9522H[n] (0x00300544 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRC1522H	15:0	0x0	RW1C	High 16 bits of the good receive packet count that are 1523-max bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

**10.2.1.17.34 Packets Transmitted (64 Bytes) Count Low - GLPRT_PTC64L[n] (0x003006A0 + 0x8*n, n=0...3)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC64L	31:0	0x0	RW1C	Low 32 bits of the transmit packet count that are 64 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.35 Packets Transmitted (64 Bytes) Count High - GLPRT_PTC64H[n] (0x003006A4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC64H	15:0	0x0	RW1C	High 16 bits of the transmit packet count that are 64 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.36 Packets Transmitted [65-127 Bytes] Count Low - GLPRT_PTC127L[n] (0x003006C0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC127L	31:0	0x0	RW1C	Low 32 bits of the transmit packet count that are 65-127 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.37 Packets Transmitted [65-127 Bytes] Count High - GLPRT_PTC127H[n] (0x003006C4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC127H	15:0	0x0	RW1C	High 16 bits of the transmit packet count that are 65-127 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.



Field	Bit(s)	Init.	Access Type	Description
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.38 Packets Transmitted [128-255 Bytes] Count Low - GLPRT_PTC255L[n]
(0x003006E0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC255L	31:0	0x0	RW1C	Low 32 bits of the transmit packet count that are 128-255 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.39 Packets Transmitted [128-255 Bytes] Count High - GLPRT_PTC255H[n]
(0x003006E4 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC255H	15:0	0x0	RW1C	High 16 bits of the transmit packet count that are 128-255 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.40 Packets Transmitted [256-511 Bytes] Count Low - GLPRT_PTC511L[n]
(0x00300700 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC511L	31:0	0x0	RW1C	Low 32 bits of the transmit packet count that are 256-511 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.41 Packets Transmitted [256-511 Bytes] Count High - GLPRT_PTC511H[n]
(0x00300704 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
PTC511H	15:0	0x0	RW1C	High 16 bits of the transmit packet count that are 256-511 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

**10.2.1.17.42 Packets Transmitted [512-1023 Bytes] Count Low - GLPRT_PTC1023L[n]
(0x00300720 + 0x8*n, n=0...3)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC1023L	31:0	0x0	RW1C	Low 32 bits of the transmit packet count that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

**10.2.1.17.43 Packets Transmitted [512-1023 Bytes] Count High - GLPRT_PTC1023H[n]
(0x00300724 + 0x8*n, n=0...3)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC1023H	15:0	0x0	RW1C	High 16 bits of the transmit packet count that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.17.44 Packets Transmitted [1024-1522 Bytes] Count Low - GLPRT_PTC1522L[n] (0x00300740 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC1522L	31:0	0x0	RW1C	Low 32 bits of the transmit packet count that are 1024-1522 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.45 Packets Transmitted [1024-1522 Bytes] Count High - GLPRT_PTC1522H[n] (0x00300744 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC1522H	15:0	0x0	RW1C	High 16 bits of the transmit packet count that are 1024-1522 bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.46 Packets Transmitted [1523-9522 bytes] Count Low - GLPRT_PTC9522L[n] (0x00300760 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC9522L	31:0	0x0	RW1C	Low 32 bits of the transmit packet count that are 1523-max bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.

10.2.1.17.47 Packets Transmitted [1523-9522 bytes] Count High - GLPRT_PTC9522H[n] (0x00300764 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PTC9522H	15:0	0x0	RW1C	High 16 bits of the transmit packet count that are 1523-max bytes (from <Destination Address> through <CRC>, inclusively) by this port. The entire register should be read by a 64-bit access.



Field	Bit(s)	Init.	Access Type	Description
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.48 Port Link XON Received Count - GLPRT_LXONRXC[n] (0x00300140 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LXONRXCNT	31:0	0x0	RW1C	Number of XON packets received by the port.

10.2.1.17.49 Port Link XOFF Received Count - GLPRT_LXOFFRXC[n] (0x00300160 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LXOFFRXCNT	31:0	0x0	RW1C	Port number of XOFF packets received by the port.

10.2.1.17.50 Port Link XON Transmitted Count - GLPRT_LXONTXC[n] (0x00300980 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LXONTXC	31:0	0x0	RW1C	Number of XON packets transmitted by the port.

10.2.1.17.51 Port Link XOFF Transmitted Count - GLPRT_LXOFFTXC[n] (0x003009A0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LXOFFTXC	31:0	0x0	RW1C	Port number of XOFF packets transmitted by the port.



10.2.1.17.52 Priority XON Received Count - GLPRT_PXONRXC[n,m] (0x00300180 + 0x8*n + 0x20*m, n=0...3, m=0...7)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRPXON RXCNT	31:0	0x0	RW1C	Number of XON packets received by the port.

10.2.1.17.53 Priority XOFF Transmitted Count - GLPRT_PXOFFTXC[n,m] (0x00300880 + 0x8*n + 0x20*m, n=0...3, m=0...7)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRPXOFF TXCNT	31:0	0x0	RW1C	Number of XOFF packets transmitted by the port.

10.2.1.17.54 Priority XON Transmitted Count - GLPRT_PXONTXC[n,m] (0x00300780 + 0x8*n + 0x20*m, n=0...3, m=0...7)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRPXONTXC	31:0	0x0	RW1C	Number of XON packets transmitted by the port.

10.2.1.17.55 Priority XOFF Received Count - GLPRT_PXOFFRXC[n,m] (0x00300280 + 0x8*n + 0x20*m, n=0...3, m=0...7)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRPXOFF RXCNT	31:0	0x0	RW1C	Number of XOFF packets received by the port.

**10.2.1.17.56 Priority XON to XOFF Count - GLPRT_RXON2OFFCNT[n,m] (0x00300380 + 0x8*n + 0x20*m, n=0...3, m=0...7)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PRRXON2OFFCNT	31:0	0x0	RW1C	Number of times transmitter transitioned from XON to XOFF of the port.

10.2.1.17.57 Port CRC Error Count - GLPRT_CRCERRS[n] (0x00300080 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
CRCERRS	31:0	0x0	RW1C	CRC error count. Counts the number of receive packets with CRC errors by the port.

10.2.1.17.58 Port Illegal Byte Error Count - GLPRT_ILLERRC[n] (0x003000E0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ILLERRC	31:0	0x0	RW1C	Counts the number of receive packets with illegal bytes errors (such as an illegal symbol in the packet) by the port.

10.2.1.17.59 Port Error Byte Count - GLPRT_ERRBC[n] (0x003000C0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ERRBC	31:0	0x0	RW1C	Counts the number of receive packets with error bytes (like an error symbol in the packet) by the port.



10.2.1.17.60 Port MAC Local Fault Count - GLPRT_MLFC[n] (0x00300020 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MLFC	31:0	0x0	RW1C	Number of faults in the local MAC by the port.

10.2.1.17.61 Port MAC Remote Fault Count - GLPRT_MRFC[n] (0x00300040 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MRFC	31:0	0x0	RW1C	Number of faults in the remote MAC.

10.2.1.17.62 Receive Length Error Count - GLPRT_RLEC[n] (0x003000A0 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RLEC	31:0	0x0	RW1C	Number of packets with receive length errors by the port. A length error occurs if an incoming packet length field in the MAC header doesn't match the packet length.

10.2.1.17.63 Receive Undersize Count - GLPRT_RUC[n] (0x00300100 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RUC	31:0	0x0	RW1C	Receive Undersize Error By The Port. This register counts the number of received frames that are shorter than minimum size (64 bytes from <Destination Address> through <CRC>, inclusively) and had a valid CRC.

**10.2.1.17.64 Receive Fragment Count - GLPRT_RFC[n] (0x00300560 + 0x8*n, n=0...3)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RFC	31:0	0x0	RW1C	Receive Fragments Count By The Port. This register counts the number of received frames that are shorter than minimum size (64 bytes from <Destination Address> through <CRC>, inclusively) and had an invalid CRC.

10.2.1.17.65 Receive Oversize Count - GLPRT_ROC[n] (0x00300120 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
ROC	31:0	0x0	RW1C	Receive Oversize Error By The Port. This register counts the number of received frames that are longer than the maximum size as defined by the Set MAC config command (from <Destination Address> through <CRC>, inclusively) and have a valid CRC.

10.2.1.17.66 Receive Jabber Count - GLPRT_RJC[n] (0x00300580 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RJC	31:0	0x0	RW1C	Number of receive jabber errors by the port. This register counts the number of received packets that passed address filtering, have a length greater than PRTGL_SAH.MFS and have bad CRC (this is slightly different from the Receive Oversize Count register). The packet's length is counted from <Destination Address> through <CRC>, inclusively.



10.2.1.17.67 Port MAC Short Packet Discard Count - GLPRT_MSPDC[n] (0x00300060 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MSPDC	31:0	0x0	RW1C	Number of MAC short packets discarded by the port.

10.2.1.17.68 Loopback Packets Discarded Count - GLPRT_LDPC[n] (0x00300620 + 0x8*n, n=0...3)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
LDPC	31:0	0x0	RW1C	Counts VM-to-VM loopback packets by the port that are dropped in the receive packet buffer. The packets are dropped due to a possible lack of bandwidth on PCIe or total bandwidth of the internal data path.

10.2.1.17.69 Switch Good Octets Received Count Low - GLSW_GORCL[n] (0x0035c000 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GORCL	31:0	0x0	RW1C	Receive Octet Count. Counts number of bytes received. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.70 Switch Good Octets Received Count High - GLSW_GORCH[n] (0x0035c004 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GORCH	15:0	0x0	RW1C	Receive Octet Count. Counts number of bytes received. Higher bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

**10.2.1.17.71 Switch Unicast Packets Received Count Low - GLSW_UPRCL[n] (0x00370000 + 0x8*n, n=0...15)**

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPRCL	31:0	0x0	RW1C	Receive Unicast Packet Count. Counts number of unicast packets received. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.72 Switch Unicast Packets Received Count High - GLSW_UPRCH[n] (0x00370004 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPRCH	15:0	0x0	RW1C	Receive Unicast Packet Count. Counts number of unicast packets received. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.73 Switch Multicast Packets Received Count Low - GLSW_MPRCL[n] (0x00370080 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPRCL	31:0	0x0	RW1C	Receive Multicast Packet Count. Counts number of multicast packets received. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.74 Switch Multicast Packets Received Count High - GLSW_MPRCH[n] (0x00370084 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPRCH	15:0	0x0	RW1C	Receive Multicast Packet Count. Counts number of multicast packets received. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.



Field	Bit(s)	Init.	Access Type	Description
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.75 Switch Broadcast Packets Received Count Low - GLSW_BPRCL[n] (0x00370100 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPRCL	31:0	0x0	RW1C	Receive Broadcast Packet Count. Counts the number of broadcast packets received. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.76 Switch Broadcast Packets Received Count High - GLSW_BPRCH[n] (0x00370104 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPRCH	15:0	0x0	RW1C	Receive Broadcast Packet Count. Counts the number of broadcast packets received. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.77 Switch Transmit Packets Discarded Count - GLSW_TDPC[n] (0x00348000 + 0x8*n, n=0...15) stats_regs

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TDPC	31:0	0x0	RW1C	Transmit Discard Packet Count.

10.2.1.17.78 Switch Received Unknown Packet Protocol Count - GLSW_RUPP[n] (0x00370180 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
RUPP	31:0	0x0	RW1C	Receive packets dropped because of an unknown protocol or no forward destination.

10.2.1.17.79 Switch Good Octets Transmit Count Low - GLSW_GOTCL[n] (0x0032C000 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GOTCL	31:0	0x0	RW1C	Transmit Octet Count. Counts the number of bytes transmitted. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.80 Switch Good Octets Transmit Count High - GLSW_GOTCH[n] (0x0032C004 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GOTCH	15:0	0x0	RW1C	Transmit Octet Count. Counts the number of bytes transmitted. Higher 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.81 Switch Unicast Packets Transmit Count Low - GLSW_UPTCL[n] (0x00340000 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPTCL	31:0	0x0	RW1C	Transmit Unicast Packet Count. Counts the number of unicast packets transmitted. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.



10.2.1.17.82 Switch Unicast Packets Transmit Count High - GLSW_UPTCH[n] (0x00340004 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPTCH	15:0	0x0	RW1C	Transmit Unicast Packet Count. Counts the number of unicast packets transmitted. Upper32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.83 Switch Multicast Packets Transmit Count Low - GLSW_MPTCL[n] (0x00340080 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPTCL	31:0	0x0	RW1C	Transmit Multicast Packet Count. Counts the number of multicast packets transmitted. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.84 Switch Multicast Packets Transmit Count High - GLSW_MPTCH[n] (0x00340084 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPTCH	15:0	0x0	RW1C	Transmit Multicast Packet Count. Counts number of multicast packets transmitted. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.85 Switch Broadcast Packets Transmit Count Low - GLSW_BPTCL[n] (0x00340100 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
BPTCL	31:0	0x0	RW1C	Transmit Broadcast Packet Count. Counts the number of broadcast packets transmitted. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.86 Switch Broadcast Packets Transmit Count High - GLSW_BPTCH[n] (0x00340104 + 0x8*n, n=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPTCH	15:0	0x0	RW1C	Transmit Broadcast Packet Count. Counts the number of broadcast packets transmitted. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.87 VEB VLAN Receive Byte Count Low - GLVEBVL_GORCL[n] (0x00360000 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLBCL	31:0	0x0	RW1C	VEB Per VLAN Byte Count Low. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.88 VEB VLAN Receive Byte Count High - GLVEBVL_GORCH[n] (0x00360004 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLBCH	15:0	0x0	RW1C	VEB Per VLAN Byte Count High. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.17.89 VEB VLAN Transmit Byte Count Low - GLVEBVL_GOTCL[n] (0x00330000 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLBCL	31:0	0x0	RW1C	VEB Per VLAN Byte Count Low. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.90 VEB VLAN Transmit Byte Count High - GLVEBVL_GOTCH[n] (0x00330004 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLBCH	15:0	0x0	RW1C	VEB Per VLAN Byte Count High. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.91 VEB VLAN Unicast Packet Count Low - GLVEBVL_UPCL[n] (0x00374000 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLUPCL	31:0	0x0	RW1C	

10.2.1.17.92 VEB VLAN Unicast Packet Count High - GLVEBVL_UPCH[n] (0x00374004 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLUPCH	15:0	0x0	RW1C	VEB Per VLAN Unicast Packet Count High. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.17.93 VEB VLAN Multicast Packet Count Low - GLVEBVL_MPCL[n] (0x00374400 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLMPCL	31:0	0x0	RW1C	VEB Per VLAN Multicast Packet Count Low. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.94 VEB VLAN Multicast Packet Count High - GLVEBVL_MPCH[n] (0x00374404 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLMPCH	15:0	0x0	RW1C	VEB Per VLAN Multicast Packet Count High. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.95 VEB VLAN Broadcast Packet Count Low - GLVEBVL_BPCL[n] (0x00374800 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLBPCL	31:0	0x0	RW1C	VEB Per VLAN Broadcast Packet Count Low. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.96 VEB VLAN Broadcast Packet Count High - GLVEBVL_BPCH[n] (0x00374804 + 0x8*n, n=0...127)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
VLBPCH	15:0	0x0	RW1C	VEB Per VLAN Broadcast Packet Count High. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.



Field	Bit(s)	Init.	Access Type	Description
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.97 VEB TC Receive Packet Count Low - GLVEBTC_RPCL[n,m] (0x00368000 + 0x8*n + 0x40*m, n=0...7, m=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCPL	31:0	0x0	RW1C	VEB Per TC Packets Count Low. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.98 VEB TC Receive Packet Count High - GLVEBTC_RPCH[n,m] (0x00368004 + 0x8*n + 0x40*m, n=0...7, m=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCPH	15:0	0x0	RW1C	VEB Per TC Packets Count High. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.99 VEB TC Receive Byte Count Low - GLVEBTC_RBCL[n,m] (0x00364000 + 0x8*n + 0x40*m, n=0...7, m=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCBCL	31:0	0x0	RW1C	VEB Per TC Byte Count Low. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.100 VEB TC Receive Byte Count High - GLVEBTC_RBCH[n,m] (0x00364004 + 0x8*n + 0x40*m, n=0...7, m=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
TCBCH	15:0	0x0	RW1C	VEB Per TC Byte Count High. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.101 VEB TC Transmit Packet Count Low - GLVEBTC_TPCL[n,m] (0x00338000 + 0x8*n + 0x40*m, n=0...7, m=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCPCL	31:0	0x0	RW1C	VEB Per TC Packets Count Low. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.102 VEB TC Transmit Packet Count High - GLVEBTC_TPCH[n,m] (0x00338004 + 0x8*n + 0x40*m, n=0...7, m=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCPCH	15:0	0x0	RW1C	VEB Per TC Packets Count High. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.103 VEB TC Transmit Byte Count Low - GLVEBTC_TBCL[n,m] (0x00334000 + 0x8*n + 0x40*m, n=0...7, m=0...15) stats_regs

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCBCL	31:0	0x0	RW1C	VEB Per TC byte Count Low. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.



10.2.1.17.104 VEB TC Transmit Byte Count High - GLVEBTC_TBCH[n,m] (0x00334004 + 0x8*n + 0x40*m, n=0...7, m=0...15)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCBCH	15:0	0x0	RW1C	VEB Per TC Byte Count High. The low and high registers are part of a 64-bit register and are read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.105 VSI Good Octets Received Count Low - GLV_GORCL[n] (0x00358000 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GORCL	31:0	0x0	RW1C	Receive Octet Count. Counts the number of bytes received by this VSI. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.106 VSI Good Octets Received Count High - GLV_GORCH[n] (0x00358004 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GORCH	15:0	0x0	RW1C	Receive Octet Count. Counts the number of bytes received by this VSI. Higher bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.



10.2.1.17.107 VSI Unicast Packets Received Count Low - GLV_UPRCL[n] (0x0036C000 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPRCL	31:0	0x0	RW1C	Receive Unicast Packet Count. Counts the number of unicast packets received by this VSI. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.108 VSI Unicast Packets Received Count High - GLV_UPRCH[n] (0x0036C004 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPRCH	15:0	0x0	RW1C	Receive Unicast Packet Count. Counts the number of unicast packets received by this VSI. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses. It is implemented internally breaking the read request into two 32-bit reads; reading the low 32-bits latches the high 32-bits into a shadow register. Reading the high 32-bit returns the value in the shadow register.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.109 VSI Multicast Packets Received Count Low - GLV_MPRCL[n] (0x0036CC00 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPRCL	31:0	0x0	RW1C	Receive Multicast Packet Count. Counts the number of multicast packets received by this VSI. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.110 VSI Multicast Packets Received Count High - GLV_MPRCH[n] (0x0036CC04 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
MPRCH	15:0	0x0	RW1C	Receive Multicast Packet Count. Counts the number of multicast packets received by this VSI. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.111 VSI Broadcast Packets Received Count Low - GLV_BPRCL[n] (0x0036D800 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPRCL	31:0	0x0	RW1C	Receive Broadcast Packet Count. Counts the number of broadcast packets received by this VSI. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.112 VSI Broadcast Packets Received Count High - GLV_BPRCH[n] (0x0036D804 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPRCH	15:0	0x0	RW1C	Receive Broadcast Packet Count. Counts the number of broadcast packets received by this VSI. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.113 VSI Received Discard Packet Count - GLV_RDPC[n] (0x00310000 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RDPC	31:0	0x0	RW1C	Counts (per VSI) packets that were dropped due to no descriptors in the host queue. For EMP VSI's, it counts dropped packets due to no EMP buffer space.



10.2.1.17.114 VSI Received Unknown Packet Protocol Count - GLV_RUPP[n] (0x0036E400 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RUPP	31:0	0x0	RW1C	Receive packets dropped because of an unknown protocol or no forward destination.

10.2.1.17.115 VSI Good Octets Transmit Count Low - GLV_GOTCL[n] (0x00328000 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GOTCL	31:0	0x0	RW1C	Transmit Octet Count. Counts number of bytes transmitted by this VSI. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.116 VSI Good Octets Transmit Count High - GLV_GOTCH[n] (0x00328004 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GOTCH	15:0	0x0	RW1C	Transmit Octet Count. Counts the number of bytes transmitted by this VSI. Higher 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.117 VSI Unicast Packets Transmit Count Low - GLV_UPTCL[n] (0x0033C000 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
UPTCL	31:0	0x0	RW1C	Transmit Unicast Packet Count. Counts the number of unicast packets transmitted by this VSI. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.



10.2.1.17.118 VSI Unicast Packets Transmit Count High - GLV_UPTCH[n] (0x0033C004 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GLVUPTCH	15:0	0x0	RW1C	Transmit Unicast Packet Count. Counts the number of unicast packets transmitted by this VSI. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.119 VSI Multicast Packets Transmit Count Low - GLV_MPTCL[n] (0x0033CC00 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPTCL	31:0	0x0	RW1C	Transmit Multicast Packet Count. Counts the number of multicast packets transmitted by this VSI. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.120 VSI Multicast Packets Transmit Count High - GLV_MPTCH[n] (0x0033CC04 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
MPTCH	15:0	0x0	RW1C	Transmit Multicast Packet Count. Counts the number of multicast packets transmitted by this VSI. Upper 32 bits. The low and high registers are part of a 64-bit register and are read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.121 VSI Broadcast Packets Transmit Count Low - GLV_BPTCL[n] (0x0033D800 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
BPTCL	31:0	0x0	RW1C	Transmit Broadcast Packet Count. Counts the number of broadcast packets transmitted by this VSI. Lower 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.

10.2.1.17.122VSI Broadcast Packets Transmit Count High - GLV_BPTCH[n] (0x0033D804 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
BPTCH	15:0	0x0	RW1C	Transmit Broadcast Packet Count. Counts the number of broadcast packets transmitted by this VSI. Upper 32 bits. The low and high registers are part of a 64-bit register and are only read using 64-bit read accesses.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.17.123VSI Transmit Error Packet Count - GLV_TEPC[n] (0x00344000 + 0x8*n, n=0...383)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TEPC	31:0	0x0	RW1C	Transmit Error Packet Count.

10.2.1.17.124Receive Error Counter 2 - GL_RXERR2_L[n] (0x0031C000 + 0x8*n, n=0...143)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RXERR2	31:0	0x0	RW1C	Count dropped packet due to one of the following exceptions (internal indication from RCU or RCB): <ul style="list-style-type: none"> • Packets directed to invalid receive queues. • Packets directed to disabled receive queues. • Packets dropped by the switch filters (these packets are counted also by the switch statistics). • Packets dropped due to MAC errors or FC CRC errors. • Packets dropped by the FD filter. • Packets dropped due to VM reset, VF reset or PF reset.



10.2.1.18 LAN Transmit Receive Registers

10.2.1.18.1 Receive Processing Block control. - GL_RDPU_CNTRL (0x00051060)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
RX_PAD_EN	0	1b	RW	Pad Rx packets with zeros that do not include CRC or the CRC is stripped to be at least 60 bytes long. Note that padding and CRC stripping for EMP packets is always done regardless of the setting of this flag.
ECO	31:1	0x0	RW	Reserved.

10.2.1.18.2 PF Queue Allocation - PFLAN_QALLOC[PF] (0x001C0400 + 0x4*PF, PF=0...15)

These registers define the LAN queue pairs allocation to the PFs. Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FIRSTQ	10:0	0x0	RW	The first LAN queue pair allocated to this PF. Valid only if the <i>Valid</i> flag is set. Valid values are 0-1535.
Reserved	15:11	0x0	Reserved	Reserved.
LASTQ	26:16	0x0	RW	The last LAN queue pair allocated to this PF. Valid only if the <i>Valid</i> flag is set. Valid values are 0-1535.
Reserved	30:27	0x0	Reserved	Reserved.
VALID	31	0b	RW	The <i>Valid</i> flag indicates that queues are allocated to this PF. For any active PF this flag must be set.

10.2.1.18.3 VF LAN Enablement - VPLAN_MAPENA[VF] (0x00074000 + 0x4*VF, VF=0...127)

GLLAN_VFENA[n] register is initialized by VFR of VF 'n'.

Reset Source: VFR

NVM Load: No

Scope: VP

Field	Bit(s)	Init.	Access Type	Description
TXRX_ENA	0	0b	RW	The VFLAN_QTABLE for the VF is enabled only when the TXRX_ENA flag is set.
Reserved	31:1	0x0	Reserved	Reserved.



10.2.1.18.4 Global RLAN Control 0 - GLLAN_RCTL_0 (0x0012A500)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PXE_MODE	0	1b	RW1C	When the PXE_MODE flag is set, the device fetches and writes back a single descriptor at a time. During normal operation (non PXE mode) this flag must be cleared.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.18.5 Global TSO TCP Mask First - GLLAN_TSOMSK_F (0x000442D8)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCPMSKF	11:0	0x0	RW	TCP flags mask for the first segment in TSO. Any bit set to 0x1 in the <i>TCPMSK</i> field clears the respective <i>TCP</i> flag in TSO. Bit 0x0 relates to the <i>FIN</i> flag. Bit 0x1 relates to the <i>SYN</i> flag and so on. Default setting: clear the <i>FIN</i> and <i>PSH</i> flags.
Reserved	31:12	0x0	Reserved	Reserved.

10.2.1.18.6 Global TSO TCP Mask Middle - GLLAN_TSOMSK_M (0x000442DC)

Reset Source: CORER.

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCPMSKM	11:0	0x0	RW	TCP flags mask for the middle segments in TSO. See <i>TCPMSKF</i> for the impact of each bit in the field. Default setting: clear the <i>FIN</i> , <i>PSH</i> and <i>CWR</i> flags.
Reserved	31:12	0x0	Reserved	Reserved.

10.2.1.18.7 Global TSO TCP Mask Last - GLLAN_TSOMSK_L (0x000442E0)

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
TCPMSKL	11:0	0x0	RW	TCP flags mask for the last segment in TSO. See <i>TCPMSKF</i> for the impact of each bit in the field. Default setting: clear the <i>CWR</i> flag.



Field	Bit(s)	Init.	Access Type	Description
Reserved	31:12	0x0	Reserved	Reserved.

10.2.1.18.8 Global Transmit Queue Control - QTX_CTL[Q] (0x00104000 + 0x4*Q, Q=0...1535)

Reset Source: CORER

NVM Load: No

Scope: Q

Field	Bit(s)	Init.	Access Type	Description
PFVF_Q	1:0	0x0	RW	Queue association to PF or VF as follows: 00b = VF queue. 01b = VM queue 10b = PF queue. 11b = Reserved.
PF_INDX	5:2	0x0	RW	PF index between 0 and 15.
Reserved	6	0b	Reserved	Reserved.
VFVM_INDX	15:7	0x0	RW	VF / VM index should be programmed per PFVF_Q setting as follows: PFVF_Q setting. Matched VF / VM index programming. 00b = VF queue. The absolute VF index (between 0-127). 01b = VM queue. The absolute VSI index (between 0-383). Else, PF or EMP queue. Must be set to zero.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.18.9 Global Transmit Pre Queue Disable - GLLAN_TXPRE_QDIS[n] (0x000e6500 + 0x4*n, n=0...11)

The access type of specific fields in this register are determined individually. It is documented per-field in the internal registers space.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
QINDX	10:0	0x0	RW	Queue index (absolute index in the device space). Each register 'n' covers 128*n to 128*n-1 queues. Software is expected to use the matched register when accessing a specific queue.
Reserved	15:11	0x0	Reserved	Reserved
Reserved	29:17	0x0	Reserved	Reserved.
SET_QDIS	30	0b	RW	Setting the SET_QDIS flag to 1b sets an internal QDIS flag of the transmit queue (that is indicated by the QINDX field in this register). As a result, any accumulated quanta of the queue are invalidated, which is a needed step before the queue is disabled. Setting the SET_QDIS flag is mutually exclusive with the CLEAR_QDIS flag.



Field	Bit(s)	Init.	Access Type	Description
CLEAR_QDIS	31	0b	RW	Setting the CLEAR_QDIS flag to 1b clears an internal QDIS flag of the transmit queue (that is indicated by the QINDX field in this register). This step should be made before the queue is enabled. Setting the CLEAR_QDIS flag is mutually exclusive with the SET_QDIS flag.

10.2.1.18.10 Global Transmit Queue Enable - QTX_ENA[Q] (0x00100000 + 0x4*Q, Q=0...1535)

The access type of specific fields in this register are determined individually. It is documented per-field in the internal registers space.

Reset Source: PFR

NVM Load: No

Scope: Q

Field	Bit(s)	Init.	Access Type	Description
QENA_REQ	0	0b	RW	Transmit Queue Enable Request. When setting this bit, software should poll the QENA_STAT flag (in this register) before using the queue. After clearing this flag, software should poll the QENA_STAT flag before releasing the memory structures. Once software changes the state of the QENA_REQ flag it must poll QENA_STAT before it is permitted to revert the state of the QENA_REQ once again.
FAST_QDIS	1	0b	RW	Fast Queue Disable. See the Fast Transmit Queue Disable Flow section for the usage of this flag.
QENA_STAT	2	0b	RO	Transmit queue enable status indication. Setting this bit to 1b indicates that the queue is active. Setting this bit to 0b, indicates that the queue is inactive.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.18.11 Global Transmit Queue Head - QTX_HEAD[Q] (0x000E4000 + 0x4*Q, Q=0...1535)

Reset Source: CORER

NVM Load: No

Scope: Q

Field	Bit(s)	Init.	Access Type	Description
Reserved	15:13	0x0	Reserved	Reserved.
Reserved	31:17	0x0	Reserved	Reserved.



10.2.1.18.12 Global Transmit Queue Tail - QTX_TAIL[Q] (0x00108000 + 0x4*Q, Q=0...1535)

Reset Source: PFR

NVM Load: No

Scope: Q

Field	Bit(s)	Init.	Access Type	Description
TAIL	12:0	0x0	RW	Transmit tail defines the first descriptor that software prepares for hardware (last valid descriptor plus one). Tail is a relative descriptor index to the beginning of the transmit descriptor ring.
Reserved	31:13	0x0	Reserved	Reserved.

10.2.1.18.13 Global Receive Queue Enable - QRX_ENA[Q] (0x00120000 + 0x4*Q, Q=0...1535)

The access type of specific fields in this register are determined individually. It is documented per-field in the internal registers space.

Reset Source: PFR

NVM Load: No

Scope: Q

Field	Bit(s)	Init.	Access Type	Description
QENA_REQ	0	0b	RW	Receive Queue Enable Request. When, setting this bit, software should poll the QENA_STAT flag (in this register) before using the queue. After clearing this flag, software should poll the QENA_STAT flag before releasing the memory structures. Once software changes the state of the QENA_REQ flag it must poll the QENA_STAT before it is permitted to revert the state of the QENA_REQ once again.
FAST_QDIS	1	0b	RW1C	Fast Queue Disable. See the Fast Receive Queue Disable Flow section for the usage of this flag. This flag is auto-cleared by hardware.
QENA_STAT	2	0b	RO	Receive Queue Enable Status Indication. Setting this bit to 1b indicates that the queue is active. Setting this bit to 0b indicates the queue is inactive.
Reserved	31:3	0x0	Reserved	Reserved.

10.2.1.18.14 Global Receive Queue Tail - QRX_TAIL[Q] (0x00128000 + 0x4*Q, Q=0...1535)

Reset Source: CORER

NVM Load: No

Scope: Q

Field	Bit(s)	Init.	Access Type	Description
TAIL	12:0	0x0	RW	Receive tail defines the first descriptor that software prepares for hardware (last valid descriptor plus one). Tail is a relative descriptor index to the beginning of the receive descriptor ring.
Reserved	31:13	0x0	Reserved	Reserved.



10.2.1.18.15 VF PF Queue Mapping Table - VPLAN_QTABLE[n,VF] (0x00070000 + 0x400*n + 0x4*VF, n=0...15, VF=0...127)

This register affects the VF but is exposed only to the parent PF.

Reset Source: VFR.

NVM Load: No

Scope: VP

Field	Bit(s)	Init.	Access Type	Description
QINDEX	10:0	0x7FF	RW	QINDEX defines the index of VF queue 'n' in the PF queues space (while 'n' is the register index). Setting QINDEX to 0x7FF means that the queue is not valid for the VF.
Reserved	31:11	0x0	Reserved	Reserved.

10.2.1.18.16 VSI Queue Control - VSILAN_QBASE[VSI] (0x0020C800 + 0x4*VSI, VSI=0...383)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: PFR

NVM Load: No

Scope: VSI

Field	Bit(s)	Init.	Access Type	Description
VSIBASE	10:0	0x0	RW	VSIBASE defines the base index of VSI 'n' within the range of the PF queues, while 'n' is the VSI register index. VSIBASE is meaningful for this VSI only if the VSIQTABLE_ENA is cleared.
VSIQTABLE_ENA	11	0b	RW	VSIQTABLE_ENA selects between a contiguous range of queues for this VSI vs. scattered range. 0b = VSI is assigned a contiguous range starting at VSIBASE. 1b = VSI is assigned a scattered range defined by VSILAN_QTABLE.
Reserved	31:12	0x0	Reserved	Reserved.

10.2.1.18.17 VSI Receive Queue Mapping Table - VSILAN_QTABLE[n,VSI] (0x00200000 + 0x800*n + 0x4*VSI, n=0...7, VSI=0...383) rcu_regs

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: PFR

NVM Load: No

Scope: VSI

Field	Bit(s)	Init.	Access Type	Description
QINDEX_0	10:0	0x0	RW	QINDEX_0 defines the index of the VSI queue '2*n' in the PF queues space (while 'n' is the register index). The absolute queue index in the device space equals to QINDEX plus PFLAN_QALLOC.FIRSTQ of the parent PF. Setting QINDEX to 0x7FF means that the queue is not valid.



Field	Bit(s)	Init.	Access Type	Description
Reserved	15:11	0x0	Reserved	Reserved.
QINDEX_1	26:16	0x0	RW	QINDEX_1 defines the index of the VSI queue '2*n+1' in the PF queues space (while 'n' is the register index). The absolute queue index in the device space equals to QINDEX plus PFLAN_QALLOC.FIRSTQ of the parent PF. Setting QINDEX to 0x7FF means that the queue is not valid.
Reserved	31:27	0x0	Reserved	Reserved

10.2.1.19 Rx Filters Registers

10.2.1.19.1 Port Queue Filter Control 0 - PRTQF_CTL_0[PRT] (0x00256E60 + 0x4*PRT, PRT=0...3)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
HSYM_ENA	0	0b	RW	Enable symmetric hash for this physical port.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.19.2 Global Queue Filter Control - GLQF_CTL (0x00269BA4)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
Reserved	0	0b	Reserved	Reserved.
HTOEP	1	0b	RW	Hash Toeplitz select for all packet types. 0b = The hash filters are based on a 32-bit XOR. 1b = The hash filters are based on the standard Toeplitz scheme while the hash key is defined per function by the xxQFHKEY registers.
PCNT_ALLOC	5:3	0x0	RW	The PCNT_ALLOC controls the GLQF_PCNT counters allocation to the PF as follows: 000b = All counters are exposed to all PFs. 100b = Each PF is allocated 1/2 of the GLQF_PCNT counters. 101b = Each PF is allocated 1/4 of the GLQF_PCNT counters. 110b = Each PF is allocated 1/8 of the GLQF_PCNT counters. 111b = Each PF is allocated 1/16 of the GLQF_PCNT counters. 001b, 010b and 011b = Reserved.



Field	Bit(s)	Init.	Access Type	Description
FD_AUTO_PCTYPE	6	0b	RW	When cleared, the PCTYPE of FD filter entries are defined by the PCTYPE field in the FD programming descriptor. When set, the PCTYPE of FD filter entries are extracted from the programming packet the same as it is done for received packets.
Reserved	7	0b	RW	Reserved.
FDBEST	24:17	0x0	RW	The FD best effort parameter defines the total number of entries in the FD table. It must not exceed the FD table size (equal to 8K) minus the sum of FDALLOC for all PFs. The global FDBEST is defined in granularity of 32 entries.
PROGPRIO	25	0b	RW	Priority ordering at filter programming between best effort space and guaranteed space. 0b = At filter programming, hardware tries first the guaranteed space. Only when it is exhausted, hardware uses the best effort space. 1b = At filter programming, hardware tries first the best effort space. Only when it is exhausted or the PF exhausted its budget in the best effort space, hardware uses the guaranteed space.
INVALPRIO	26	0b	RW	Priority ordering at filter invalidation between best effort space and guaranteed space. 0b = At filter invalidation, hardware tries first to increment the best effort space. Only when the global best effort space is at its maximum value or the best effort space of the PF is at its maximum value, the guaranteed space is incremented. 1b = At filter invalidation, hardware tries first to increment its guaranteed space. Only when it is already at its maximum value, the best effort space is incremented.
Reserved	31:28	0x0	Reserved	Reserved.

10.2.1.19.3 PF Queue Filter Control 1 - PFQF_CTL_1[PF] (0x00245D80 + 0x4*PF, PF=0...15)

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
CLEARFD TABLE	0	0b	RW1C	Setting the <i>Clear Table</i> flag by software, hardware invalidates all entries of the PF in the FD table. Once all entries of the PF are invalidated, the <i>Clear Table</i> flag is cleared as well.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.19.4 Global Queue Filter SWAP Fields - GLQF_SWAP[n,m] (0x00267E00 + 0x4*n + 0x8*m, n=0...1, m=0...63)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: Raw Data

Scope: GL



Field	Bit(s)	Init.	Access Type	Description
OFF0_SRC0	5:0	0x0	RW	Offset of the first field of the first couple in the field vector to be swapped. The offset is defined in word units.
OFF0_SRC1	11:6	0x0	RW	Offset of the second field of the first couple in the field vector to be swapped. The offset is defined in word units.
FLEN0	15:12	0x0	RW	Field length in word units. When FLEN0 is set to zero, the fields defined by OFF0 are not candidates for swapping.
OFF1_SRC0	21:16	0x0	RW	Offset of the first field of the second couple in the field vector to be swapped. The offset is defined in word units.
OFF1_SRC1	27:22	0x0	RW	Offset of the second field of the second couple in the field vector to be swapped. The offset is defined in word units.
FLEN1	31:28	0x0	RW	Field length in word units. When the FLEN1 is set to zero, the fields defined by OFF1 are not candidates for swapping.

10.2.1.19.5 Global Queue Filter Symmetric Hash Enablement - GLQF_HSYM[n] (0x00269D00 + 0x4*n, n=0...63)

These registers allow manipulations over the type rule bus. The changes applied on the type rule bus are used by the L2 filters and has no affect on the Q selection by type rule.

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
SYMH_ENA	0	0b	RW	Enables symmetric hash for PCTYPE 'n', while 'n' is the register index.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.19.6 Port Queue Filter - Flexible Parser Information Table - PRTQF_FLX_PIT[n,PRT] (0x00255200 + 0x20*n + 0x4*PRT, n=0...8, PRT=0...3)

These registers define the byte stream extracted to the field vector (unique setting option per LAN port).

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
SOURCE_OFF	4:0	0x0	RW	Source word offset in the mapped protocol layer header starting from its beginning. When setting source offset rules: <ul style="list-style-type: none"> The SOURCE_OFF plus FSIZE should not exceed byte 480 of the packet. Must be programmed in ascending order: current offset ≥ previous offset + previous FSIZE. The previous rule applies for all entries, including non-used ones.



Field	Bit(s)	Init.	Access Type	Description
FSIZE	9:5	0x0	RW	Field size defined in word units. For non-used registers, FSIZE must be set to 0x1.
DEST_OFF	15:10	0x0	RW	Destination word offset in the field vector. The destination offset can be set to 50...57 matching offset 0...7 in the flexible field vector, respectively. While DEST_OFF plus FSIZE must not be greater than 58. For non-used registers, DEST_OFF must be set to 63 (outside the range for active entries).
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.19.7 Port Queue Filter Flow Director Flexible Input Set - PRTQF_FD_FLXINSET[n,PRT] (0x00253800 + 0x20*n + 0x4*PRT, n=0...63, PRT=0...3) rcu_regs

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
INSET	7:0	0x0	RW	Bit 'i' of the INSET enables word 57 minus 'i' in the field vector. It is mapped to bits 50:57 of the PRTQF_FD_INSET (equals to bits 6:13 in the PRTQF_FD_INSET[m=1] -> INSET, respectively).
Reserved	31:8	0x0	Reserved	Reserved.

10.2.1.19.8 Port Queue Filter Flow Director Mask - PRTQF_FD_MSK[n,m,PRT] (0x00252000 + 0x40*n + 0x20*m + 0x4*PRT, n=0...63, m=0...1, PRT=0...3)

There are two global mask registers ('n' = 0, 1) for each packet type ('m' = 0,... 63). Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MASK	15:0	0x0	RW	Each bit set to 0x1 disables the matched bit in the selected word within the field vector defined by the OFFSET field.
OFFSET	21:16	0x0	RW	Word offset in the field vector. OFFSET must be in the range 50 to 57 for words 0 to 7, respectively.
Reserved	31:22	0x0	Reserved	Reserved.



10.2.1.19.9 PF Queue Filter Hash Enabled Packet Type - PFQF_HENA[n,PF] (0x00245900 + 0x80*n + 0x4*PF, n=0...1, PF=0...15)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
PTYPE_ENA	31:0	0x0	RW	Packet type enablement of the hash filter for the function. Bit 'm' in register 'n' enables packet type '32 x n + m' as defined in the Packet Types For Classification Filters table.

10.2.1.19.10 VF Queue Filter Hash Enabled Packet Type - VFQF_HENA[n,VF] (0x00230800 + 0x400*n + 0x4*VF, n=0...1, VF=0...127)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
PTYPE_ENA	31:0	0x0	RW	Packet type enablement of the hash filter for the function. Bit 'm' in register 'n' enables packet type '32 x n + m' as defined in the Packet Types For Classification Filters table.

10.2.1.19.11 PF Queue Filter Hash Region of Queues - PFQF_HREGION[n,PF] (0x00245400 + 0x80*n + 0x4*PF, n=0...7, PF=0...15)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
OVERRIDE_ENA_0	0	0b	RW	Override traffic class region for packet type '8 x n' while 'n' is the register index.
REGION_0	3:1	0x0	RW	Receive queue region for packet type '8 x n' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_1	4	0b	RW	Override traffic class region for packet type '8 x n + 1' while 'n' is the register index.



Field	Bit(s)	Init.	Access Type	Description
REGION_1	7:5	0x0	RW	Receive queue region for packet type '8 x n + 1' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_2	8	0b	RW	Override traffic class region for packet type '8 x n + 2' while 'n' is the register index.
REGION_2	11:9	0x0	RW	Receive queue region for packet type '8 x n + 2' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_3	12	0b	RW	Override traffic class region for packet type '8 x n + 3' while 'n' is the register index.
REGION_3	15:13	0x0	RW	Receive queue region for packet type '8 x n + 3' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_4	16	0b	RW	Override traffic class region for packet type '8 x n + 4' while 'n' is the register index.
REGION_4	19:17	0x0	RW	Receive queue region for packet type '8 x n + 4' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_5	20	0b	RW	Override traffic class region for packet type '8 x n + 5' while 'n' is the register index.
REGION_5	23:21	0x0	RW	Receive queue region for packet type '8 x n + 5' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_6	24	0b	RW	Override traffic class region for packet type '8 x n + 6' while 'n' is the register index.
REGION_6	27:25	0x0	RW	Receive queue region for packet type '8 x n + 6' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_7	28	0b	RW	Override traffic class region for packet type '8 x n + 7' while 'n' is the register index.
REGION_7	31:29	0x0	RW	Receive queue region for packet type '8 x n + 7' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.



10.2.1.19.12 VF Queue Filter Hash Region of Queues - VFQF_HREGION[n,VF] (0x0022E000 + 0x400*n + 0x4*VF, n=0...7, VF=0...127)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	CFG Policy	Description
OVERRIDE_ENA_0	0	0b	RW	UNDEFINED	Override traffic class region for packet type '8 x n' while 'n' is the register index.
REGION_0	3:1	0x0	RW	UNDEFINED	Receive queue region for packet type '8 x n' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_1	4	0b	RW	UNDEFINED	Override traffic class region for packet type '8 x n + 1' while 'n' is the register index.
REGION_1	7:5	0x0	RW	UNDEFINED	Receive queue region for packet type '8 x n + 1' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_2	8	0b	RW	UNDEFINED	Override traffic class region for packet type '8 x n + 2' while 'n' is the register index.
REGION_2	11:9	0x0	RW	UNDEFINED	Receive queue region for packet type '8 x n + 2' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_3	12	0b	RW	UNDEFINED	Override traffic class region for packet type '8 x n + 3' while 'n' is the register index.
REGION_3	15:13	0x0	RW	UNDEFINED	Receive queue region for packet type '8 x n + 3' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_4	16	0b	RW	UNDEFINED	Override traffic class region for packet type '8 x n + 4' while 'n' is the register index.
REGION_4	19:17	0x0	RW	UNDEFINED	Receive queue region for packet type '8 x n + 4' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_5	20	0b	RW	UNDEFINED	Override traffic class region for packet type '8 x n + 5' while 'n' is the register index.
REGION_5	23:21	0x0	RW	UNDEFINED	Receive queue region for packet type '8 x n + 5' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_6	24	0b	RW	UNDEFINED	Override traffic class region for packet type '8 x n + 6' while 'n' is the register index.
REGION_6	27:25	0x0	RW	UNDEFINED	Receive queue region for packet type '8 x n + 6' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_7	28	0b	RW	UNDEFINED	Override traffic class region for packet type '8 x n + 7' while 'n' is the register index.



Field	Bit(s)	Init.	Access Type	CFG Policy	Description
REGION_7	31:29	0x0	RW	UNDEFINED	Receive queue region for packet type '8 x n + 7' while 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.

10.2.1.19.13 VSI Receive Traffic Class Queues - VSIQF_TCREGION[n,VSI] (0x00206000 + 0x800*n + 0x4*VSI, n=0...3, VSI=0...383)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: PFR

NVM Load: No

Scope: VSI

Field	Bit(s)	Init.	Access Type	Description
TC_OFFSET	8:0	0x0	RW	Relative queue index to the beginning of the VSI that the TC '2*n' uses. While 'n' is the register index. Note that hardware does not check if the queue index exceeds the VSI range. So it is the PF software responsibility to make sure that TC_SIZE + TC_OFFSET does not exceeds the VSI range.
TC_SIZE	11:9	0x0	RW	The number of receive queues allocated to TC '2*n' for the VSI. While 'n' is the register index. The TCSIZE can have any value of 0 to 6, which means any of the following respective number of allocated queues: 1, 2, 4, 8, 16, 32 and 64. Note: 7 is a reserved value.
Reserved	15:12	0x0	Reserved	Reserved.
TC_OFFSET2	24:16	0x0	RW	Relative queue index to the beginning of the VSI that the TC '2*n+1' uses. While 'n' is the register index. Note that hardware does not check if the queue index exceeds the VSI range. So it is the PF software responsibility to make sure that TC_SIZE + TC_OFFSET does not exceeds the VSI range.
TC_SIZE2	27:25	0x0	RW	The number of receive queues allocated to TC '2*n+1' for the VSI. While 'n' is the register index. The TCSIZE can have any value of 0 to 6, which means any of the following respective number of allocated queues: 1, 2, 4, 8, 16, 32 and 64. Note: 7 is a reserved value.
Reserved	31:28	0x0	Reserved	Reserved.

10.2.1.19.14 PF Queue Filter Hash Key - PFQF_HKEY[n,PF] (0x00244800 + 0x80*n + 0x4*PF, n=0...12, PF=0...15)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
KEY_0	7:0	0x0	RW	Toeplitz key byte '4*n+0' while 'n' is the register index.
KEY_1	15:8	0x0	RW	Toeplitz key byte '4*n+1' while 'n' is the register index.



Field	Bit(s)	Init.	Access Type	Description
KEY_2	23:16	0x0	RW	Toeplitz key byte '4*n+2' while 'n' is the register index.
KEY_3	31:24	0x0	RW	Toeplitz key byte '4*n+3' while 'n' is the register index.

10.2.1.19.15 VF Queue Filter Hash Key - VFQF_HKEY[n,VF] (0x00228000 + 0x400*n + 0x4*VF, n=0...12, VF=0...127)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
KEY_0	7:0	0x0	RW	Toeplitz key byte '4*n+0' while 'n' is the register index.
KEY_1	15:8	0x0	RW	Toeplitz key byte '4*n+1' while 'n' is the register index.
KEY_2	23:16	0x0	RW	Toeplitz key byte '4*n+2' while 'n' is the register index.
KEY_3	31:24	0x0	RW	Toeplitz key byte '4*n+3' while 'n' is the register index.

10.2.1.19.16 Global Queue Filter Hash Key - GLQF_HKEY[n] (0x00270140 + 0x4*n, n=0...12)

Reset Source: CORER.

NVM Load: Type1/2 in CORER register auto-load module

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
KEY_0	7:0	0x0	RW	Toeplitz key byte '4*n+0' while 'n' is the register index.
KEY_1	15:8	0x0	RW	Toeplitz key byte '4*n+1' while 'n' is the register index.
KEY_2	23:16	0x0	RW	Toeplitz key byte '4*n+2' while 'n' is the register index.
KEY_3	31:24	0x0	RW	Toeplitz key byte '4*n+3' while 'n' is the register index.

10.2.1.19.17 PF Queue Filter Hash LUT - PFQF_HLUT[n,PF] (0x00240000 + 0x80*n + 0x4*PF, n=0...127, PF=0...15)

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
LUT0	5:0	0x0	RW	Hash redirection LUT entry 4 x 'n' while 'n' is the register index.
Reserved	7:6	0x0	Reserved	Reserved.
LUT1	13:8	0x0	RW	Hash redirection LUT entry 4 x 'n' + 1 while 'n' is the register index.
Reserved	15:14	0x0	Reserved	Reserved.



Field	Bit(s)	Init.	Access Type	Description
LUT2	21:16	0x0	RW	Hash redirection LUT entry 4 x 'n' + 2 while 'n' is the register index.
Reserved	23:22	0x0	Reserved	Reserved.
LUT3	29:24	0x0	RW	Hash redirection LUT entry 4 x 'n' + 3 while 'n' is the register index.
Reserved	31:30	0x0	Reserved	Reserved.

10.2.1.19.18 VF Queue Filter Hash LUT - VFQF_HLUT[n,VF] (0x00220000 + 0x400*n + 0x4*VF, n=0...15, VF=0...127)

Reset Source: CORER

NVM Load: No

Scope: VF

Field	Bit(s)	Init.	Access Type	Description
LUT0	3:0	0x0	RW	Hash redirection LUT entry 4 x 'n' while 'n' is the register index.
Reserved	7:4	0x0	Reserved	Reserved.
LUT1	11:8	0x0	RW	Hash redirection LUT entry 4 x 'n' + 1 while 'n' is the register index.
Reserved	15:12	0x0	Reserved	Reserved.
LUT2	19:16	0x0	RW	Hash redirection LUT entry 4 x 'n' + 2 while 'n' is the register index.
Reserved	23:20	0x0	Reserved	Reserved.
LUT3	27:24	0x0	RW	Hash redirection LUT entry 4 x 'n' + 3 while 'n' is the register index.
Reserved	31:28	0x0	Reserved	Reserved.

10.2.1.19.19 Global Queue Filter Packet Counter - GLQF_PCNT[n] (0x00266800 + 0x4*n, n=0...511)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
PCNT	31:0	0x0	RW1C	Packet counter indicated by the FD filter(s). Note that the counter saturates at 0xFF..F. This register is RWC rather than RW1C. Writing any data clears the entire register.

10.2.1.19.20 Global Queue Filter Flow Director Status 0 - GLQF_FDCNT_0 (0x00269BAC)

Reset Source: CORER

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
GUARANT_CNT	12:0	0x0	RO	Total number of FD entries in guaranteed spaces of all PFs.



Field	Bit(s)	Init.	Access Type	Description
BESTCNT	25:13	0x0	RO	Total number of FD entries in the best effort space.
Reserved	31:26	0x0	Reserved	Reserved.

10.2.1.19.21 PF Queue Filter Flow Director Allocation - PFQF_FDALLOC[PF] (0x00246280 + 0x4*PF, PF=0...15)

Accessing this register should be done by the Receive Control CSR Read / Write admin command rather than direct access.

Reset Source: CORER

NVM Load: Type1/2 in CORER register auto-load module

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
FDALLOC	7:0	0x0	RW	The FD allocation parameter defines the number of guaranteed entries in the FD table. It is defined in granularity of 32 entries.
FDBEST	15:8	0x0	RW	The FD best effort parameter defines the maximum number of entries the PF can consume from the shared space. It is defined in granularity of 32 entries.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.19.22 PF Queue Filter Flow Director Allocation Status - PFQF_FDSTAT[PF] (0x00246380 + 0x4*PF, PF=0...15)

This register counts the number of FD filters of the PF. Note that following a PFR, all the filters of the PF are removed and the counter is decremented to zero.

Reset Source: CORER

NVM Load: No

Scope: PF

Field	Bit(s)	Init.	Access Type	Description
GUARANT_CNT	12:0	0x0	RO	GUARANT_CNT indicates the number of FD entries consumed by the PF out of its guaranteed space.
Reserved	15:13	0x0	Reserved	Reserved.
BEST_CNT	28:16	0x0	RO	BEST_CNT indicates the number of FD entries consumed by the PF out of its best effort space.
Reserved	31:29	0x0	Reserved	Reserved.

10.2.1.20 TimeSync (IEEE 1588) Registers

10.2.1.20.1 Port Time Sync Control 0 - PRRTSYN_CTL0[PRT] (0x001E4200 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: Type1/2 in GLOBR register auto-load module

Scope: PRT



Field	Bit(s)	Init.	Access Type	Description
TXTIME_INT_ENA	1	0b	RW	Interrupt enable when the TSYNXTIME registers samples the transmission time in this port. The event is reported in the TXTIME bit in the PRRTSYN_STAT_0 register.
EVENT_INT_ENA	2	0b	RW	Interrupt enable when an event is sampled in any of the PRRTSYN_EVT registers.
TGT_INT_ENA	3	0b	RW	Interrupt enable when the target time is sampled in any of the PRRTSYN_TGT registers.
Reserved	7:4	0x0	Reserved	Reserved.
PF_ID	11:8	0x0	RW	Software indication for the PF function ID that controls the 1588 logic of the port (no hardware impact). This field is expected to be loaded from the NVM or set by the management agent. During normal operation, the PF software driver is not expected to change its setting.
TSYNACT	13:12	0x0	RW	Software indication for 1588 mode of operation (no hardware impact): 00b = Inactive agent. 01b = Synchronized to local time. 10b = Synchronized to the standard TAI. 11b = Reserved.
Reserved	30:14	0x0	Reserved	Reserved
TSYNENA	31	0b	RW	Enable 1588 Logic. When the TSYNENA flag is cleared, 1588 logic is not functional. This flag must be set the same as the TSYNENA flag in the PRRTSYN_CTL1 register.

10.2.1.20.2 Port Time Sync Control 1 - PRRTSYN_CTL1[PRT] (0x00085020 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
V1MESS_TYPE0	7:0	0x01	RW	PTP V1 message type 0 for sampled timestamp of received 1588 packets. The default setting is 0x01 for Sync and Delay_Req packets. Setting this field to 0xFF is like a wild card option enabling all PTP V1 packets.
V1MESS_TYPE1	15:8	0x01	RW	PTP V1 message type 1 for sampled timestamp of received 1588 packets. The default setting is 0x01 for Sync and Delay_Req packets. Setting this field to 0xFF is like a wild card option enabling all PTP V1 packets.
V2MESS_TYPE0	19:16	0x0	RW	PTP V2 message 0 type for sampled timestamp of received 1588 packets. The default setting is 0x0 for Sync packets. Other values are: 0x1 = Delay request. 0x2 = Pdelay request. 0x3 =Pdelay Response. Setting this field to 0xF is like a wild card option enabling all PTP V2 packets.
V2MESS_TYPE1	23:20	0x1	RW	PTP V2 message 1 type for sampled timestamp of received 1588 packets. The default setting is 0x1 for delay request.



Field	Bit(s)	Init.	Access Type	Description
TSYNTYPE	25:24	0x0	RW	Receive packets types sampled by the 1588 timer. 00b = L2 version 2 packets (<i>Message Type</i> is defined by the PRTTSYN_CTL1 register). 01b = UDP version 1 packets (UDP ports are enabled by UDP_ENA field in this register and <i>Message Type</i> field is defined by the PRTTSYN_CTL1 register). 10b = L2 and UDP version 2 packets (UDP ports are enabled by UDP_ENA field in this register and <i>Message Type</i> is defined by the PRTTSYN_CTL1 register). 11b = L2 and UDP version 2 event packets (UDP ports are enabled by UDP_ENA field in this register and <i>Message Type</i> < 8).
UDP_ENA	27:26	0x0	RW	Enable the UDP ports recognized as 1588 packets. 00b = No UDP packet recognition. 01b = UDP port number equals to 0x013F. 10b = UDP port number equals to 0x0140. 11b = UDP port numbers equals to either 0x013F or 0x140.
Reserved	30:28	0x0	Reserved	Reserved.
TSYNENA	31	0b	RW	Enable 1588 Logic. When the <i>TSYNENA</i> flag is cleared, 1588 logic is not functional. This flag must be set the same as the <i>TSYNENA</i> flag in the PRTTSYN_CTL0 register.

10.2.1.20.3 Port Time Sync Status 0 - PRTTSYN_STAT_0[PRT] (0x001E4220 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
EVENT0	0	0b	RCW	Set to 1b when PRTTSYN_EVNT[0] captures an input event timestamp.
EVENT1	1	0b	RCW	Set to 1b when PRTTSYN_EVNT[1] captures an input event timestamp.
TGT0	2	0b	RCW	Set to 1b when the PRTTSYN_TGT[0] timer expired.
TGT1	3	0b	RCW	Set to 1b when the PRTTSYN_TGT[1] timer expired.
TXTIME	4	0b	RCW	Set to 1b when the PRTTSYN_TXTIME register samples a Tx packet.
Reserved	31:5	0x0	Reserved	Reserved.

10.2.1.20.4 Port Time Sync Status 1 - PRTTSYN_STAT_1[PRT] (0x00085140 + 0x4*PRT, PRT=0...3)

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
RXT0	0	0b	RO	The PRTTSYN_RXTIME[0] register contains a valid timestamp. This bit is set by hardware when received packet reception time is captured in the PRTTSYN_RXTIME[0] register and it is auto-cleared when software reads the PRTTSYN_RXTIME[0] register.



Field	Bit(s)	Init.	Access Type	Description
RXT1	1	0b	RO	The PRRTSYN_RXTIME[1] register contains a valid timestamp. This bit is set by hardware when received packet reception time is captured in the PRRTSYN_RXTIME[1] register and it is auto-cleared when software reads the PRRTSYN_RXTIME[1] register.
RXT2	2	0b	RO	The PRRTSYN_RXTIME[2] register contains a valid timestamp. This bit is set by hardware when received packet reception time is captured in the PRRTSYN_RXTIME[2] register and it is auto-cleared when software reads the PRRTSYN_RXTIME[2] register.
RXT3	3	0b	RO	The PRRTSYN_RXTIME[3] register contains a valid timestamp. This bit is set by hardware when received packet reception time is captured in the PRRTSYN_RXTIME[3] register and it is auto-cleared when software reads the PRRTSYN_RXTIME[3] register.
Reserved	31:4	0x0	Reserved	Reserved.

10.2.1.20.5 Port Time Sync Time Low - PRRTSYN_TIME_L[PRT] (0x001E4100 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNTIME_L	31:0	0x0	RW	Bits 32...63 of the 96-bit timer. If the lowest 32 bits define the fraction of ns then this register defines the lower 32 bits of the ns units.

10.2.1.20.6 Port Time Sync Time High - PRRTSYN_TIME_H[PRT] (0x001E4120 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNTIME_H	31:0	0x0	RW	Upper 32 bits of the 96-bit timer.

10.2.1.20.7 Port Time Sync Increment Value Low - PRRTSYN_INC_L[PRT] (0x001E4040 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNINC_L	31:0	0x0	RW	32 LS bits of the increment value added to the 96-bit TSYNTIME registers at each MAC clock.



10.2.1.20.8 Port Time Sync Increment Value High - PRRTSYN_INC_H[PRT] (0x001E4060 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNINC_H	5:0	0x0	RW	6 MS bits of the increment value added to the 96-bit TSYNTIME registers at each MAC clock.
Reserved	31:6	0x0	Reserved	Reserved

10.2.1.20.9 Port Time Sync Adjustment - PRRTSYN_ADJ[PRT] (0x001E4280 + 0x4*PRT, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNADJ	30:0	0x0	RW	Absolute value of the time adjust matched to the units of the TSYN_TIME_L register.
SIGN	31	0b	RW	Time Adjustment Sign. 0b = Positive adjustment. 1b = Negative adjustment

10.2.1.20.10 Port Time Sync Receive PTP Packet Time Low - PRRTSYN_RXTIME_L[n,PRT] (0x000850C0 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
RXTIME_L	31:0	0x0	RO	32 LS bits of the receive PTP packet time matches the units of the TSYN_TIME_L register.

10.2.1.20.11 port Time Sync Receive PTP Packet Time High - PRRTSYN_RXTIME_H[n,PRT] (0x00085040 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

Reset Source: CORER

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
RXTIME_H	31:0	0x0	RO	32 MS bits of the receive PTP packet time matches the units of the TSYN_TIME_H register.

**10.2.1.20.12 Port Time Sync Transmit Packet Time Low - PRTTSYN_TXTIME_L[PRT]
(0x001E41C0 + 0x4*PRT, PRT=0...3)**

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TXTIME_L	31:0	0x0	RO	32 LS bits of the sampled 1588 time of a Tx packet matches the units of the TSYN_TIME_L register.

**10.2.1.20.13 Port Time Sync Transmit Packet Time High - PRTTSYN_TXTIME_H[PRT]
(0x001E41E0 + 0x4*PRT, PRT=0...3)**

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TXTIME_H	31:0	0x0	RO	32 MS bits of the sampled 1588 time of a Tx packet matches the units of the TSYN_TIME_H register.

**10.2.1.20.14 Port Time Sync AUX Control 0 - PRTTSYN_AUX_0[n,PRT] (0x001E42A0 +
0x20*n + 0x4*PRT, n=0...1, PRT=0...3)**

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
OUT_ENA	0	0b	RW	Synchronized Output Enablement. When set to 1b, the synchronized output signal is enabled according to the other parameters in this register.
OUTMOD	2:1	0x0	RW	Output Signal Operation Mode. 00b = Output level mode. 01b = Flipped output mode. 10b = Output pulse mode. 11b = Output clock mode. The GPIO signals should be set as a 1588 output by the GLGEN_GPIO_CTL[n] registers.
OUTLVL	3	0b	RW	Output level driven on the I/O signal at the target time.
Reserved	7:4	0x0	Reserved	Reserved.
PULSEW	11:8	0x0	RW	Output pulse width for output pulse mode equals to 16 x (PULSEW + 1) clocks. The clock frequency is defined per-link speed in the 1588 Clock Registers section.
Reserved	15:12	0x0	Reserved	Reserved.



Field	Bit(s)	Init.	Access Type	Description
EVNTLVL	17:16	0x0	RW	Event level on the I/O signal configured as a 1588 input. It can be set to one of the following options: 00b = Disable. 01b = Rising edge. 10b = Falling edge. 11b = Any transition. The GPIO signals should be set as a 1588 input by the GLGEN_GPIO_CTL[n] registers.
Reserved	31:18	0x0	Reserved	Reserved.

10.2.1.20.15 Port Time Sync AUX Control 1 - PRTTSYN_AUX_1[n,PRT] (0x001E42E0 + 0x20*n + 0x4*PRT, n=0...1, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
INSTNT	0	0b	RW	When setting the <i>INSTNT</i> flag, the OUTSIG signal is forced to the OUTLVL value. This flag is auto-cleared by hardware.
SAMPLE_TIME	1	0b	RW	Setting the SAMPLE_TIME flag triggers instant sampling of the PRTTSYN_TIME to the matched PRTTSYN_EVNT register. This flag is auto-cleared by hardware.
Reserved	31:2	0x0	Reserved	Reserved.

10.2.1.20.16 Port Time Sync Target Time Low - PRTTSYN_TGT_L[n,PRT] (0x001E4140 + 0x20*n + 0x4*PRT, n=0...1, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNTGTT_L	31:0	0x0	RW	32 LS bits of the target time of an event out in one of the AUX I/O signals.

10.2.1.20.17 Port Time Sync Target Time High - PRTTSYN_TGT_H[n,PRT] (0x001E4180 + 0x20*n + 0x4*PRT, n=0...1, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNTGTT_H	31:0	0x0	RW	32 MS bits of the target time of an event out in one of the AUX I/O signals.

**10.2.1.20.18 Port Time Sync Clock Out Duration - PRTTSYN_CLKO[n,PRT] (0x001E4240 + 0x20*n + 0x4*PRT, n=0...1, PRT=0...3)**

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNCKO	31:0	0x0	RW	Clock output duration as described in the Auxiliary 1588 I/O signals section.

10.2.1.20.19 Port Time Sync Event Time Low - PRTTSYN_EVNT_L[n,PRT] (0x001E4080 + 0x20*n + 0x4*PRT, n=0...1, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNEVNT_L	31:0	0x0	RO	32 LS bits of the sampled event time. The sampled event is defined by the <i>EVNTLVL</i> field in the PRTTSYN_AUX register.

10.2.1.20.20 Port Time Sync Event Time High - PRTTSYN_EVNT_H[n,PRT] (0x001E40C0 + 0x20*n + 0x4*PRT, n=0...1, PRT=0...3)

Reset Source: GLOBR

NVM Load: No

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
TSYNEVNT_H	31:0	0x0	RO	32 MS bits of the sampled event time of a 1588 event defined by the PRTTSYN_AUX register.

10.2.1.21 Manageability Registers**10.2.1.21.1 Firmware Reset Count - GL_FWRESETCNT (0x00083100)**

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FWRESETCNT	31:0	0x0	RO	Firmware Resets Count. Updated by hardware and saturates at 0xFFFF,FFFF.



10.2.1.21.2 Management Control Register - PRT_MNG_MANC[PRT] (0x00256A20 + 0x4*PRT, PRT=0...3)

The MANC register can be written by the MC and is not accessible to the host for writing.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
FLOW_CONTROL_DISCARD	0	0b	RW	0b = Apply filtering rules to packets with FC EtherType. 1b = Discard packets with FC EtherType. Note: FC EtherType is 0x8808.
NCSI_DISCARD	1	0b	RW	0b = Apply filtering rules to packets with NC-SI EtherType. 1b = Discard packets with NC-SI EtherType. Note: NC-SI EtherType is 0x88F8.
Reserved	16:2	0x0	Reserved	Reserved.
RCV_TCO_EN	17	0b	RW	Receive TCO Packets Enabled. When this bit is set it enables the receive flow to the manageability block. This bit should be set only if at least one of MANC.EN_BMC2OS or MANC.EN_BMC2NET bits are set.
Reserved	18	0b	Reserved	Reserved.
Reserved	24:20	0x0	Reserved	Reserved.
FIXED_NET_TYPE	25	0b	RW	Fixed Next Type. If set, only packets matching the net type defined by the NET_TYPE field passes to manageability. Otherwise, both tagged and un-tagged packets might be forwarded to the manageability block.
NET_TYPE	26	0b	RW	NET TYPE. 0b = Pass only un-tagged packets. 1b = Pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE is set.
Reserved	27	0b	Reserved	Reserved.
EN_BMC2OS	28	0b	RW	Enable MC to operating system and operating system to MC traffic. 0b = The MC cannot communicate with the operating system. 1b = The MC can communicate with the operating system. When cleared, the MC traffic is not forwarded to the operating system even if host address filtering indicates that it should. When cleared, operating system traffic is not forwarded to the MC even if the manageability decision filters indicate it should. This bit does not impact the MC-to-network traffic. Notes: 1. Initial value loaded according to value of port 'n' traffic types field in the NVM. 2. Bit reflects the internal management Aux register bit.



Field	Bit(s)	Init.	Access Type	Description
EN_BMC2NET	29	0b	RW	<p>Enable the MC to network and network to MC traffic. 0b = The MC cannot communicate with the network. 1b = The MC can communicate with the network When cleared, MC traffic is not forwarded to the network and the network traffic is not forwarded to the MC even if the decision filters indicates it should. This bit does not impact the host-to-MC traffic.</p> <p>Notes:</p> <ol style="list-style-type: none"> Initial value loaded according to value of port 'n' traffic types field in the NVM. Bit reflects the internal management Aux register bit. This bit can change while the host is sending or receiving traffic.
Reserved	31:30	0x0	Reserved	Reserved.

10.2.1.21.3 Manageability Decision Filters - PRT_MNG_MDEF_EXT[n,PRT] (0x00255F00 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
L2_ETHERTYPE_AND	3:0	0x0	RW	L2 EtherType. Controls the inclusion of L2 EtherType filtering in the manageability filter decision (AND section).
L2_ETHERTYPE_OR	7:4	0x0	RW	L2 EtherType. Controls the inclusion of L2 EtherType filtering in the manageability filter decision (OR section).
FLEX_PORT_OR	23:8	0x0	RW	Flex Port. Controls the inclusion of flex port filtering in the manageability filter decision (OR section). Bit 16 corresponds to flex port 0, etc.
FLEX_TCO	24	0b	RW	Flex TCO. Controls the inclusion of flex TCO filtering in the manageability filter decision (OR section). Bit 24 corresponds to the flex TCO filter. Note: Supported only for network traffic.
NEIGHBOR_DISCOVERY_135_OR	25	0b	RW	Neighbor Discovery. Controls the inclusion of neighbor discovery filtering in the manageability filter decision (OR section). The neighbor type accepted by this filter is type 0x87 (135).
NEIGHBOR_DISCOVERY_136_OR	26	0b	RW	Neighbor Discovery. Controls the inclusion of neighbor discovery filtering in the manageability filter decision (OR section). The neighbor type accepted by this filter is type 0x88 (136).
NEIGHBOR_DISCOVERY_137_OR	27	0b	RW	Neighbor Discovery. Controls the inclusion of neighbor discovery filtering in the manageability filter decision (OR section). The neighbor type accepted by this filter is type 0x89 (137).
ICMP_OR	28	0b	RW	Controls the inclusion of ICMP filtering in the manageability filter decision (OR section).
MLD	29	0b	RW	MLD. Controls the inclusion of MLD packets. These are ICMPv6 packets with the following types: 130, 131, 132, 143.
APPLY_TO_NETWORK_TRAFFIC	30	0b	RW	0b = This decision filter does not apply to traffic received from the network. 1b = This decision filter applies to traffic received from the network.
APPLY_TO_HOST_TRAFFIC	31	0b	RW	0b = This decision filter does not apply to traffic received from the host. 1b = This decision filter applies to traffic received from the host.



10.2.1.21.4 Manageability Decision Filters1 - PRT_MNG_MDEF[n,PRT] (0x00255D00 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MAC_EXACT_AND	3:0	0x0	RW	Exact. Controls the inclusion of the exact MAC address 0 to 3. in the manageability filter decision (AND section). Bit 0 corresponds to the exact MAC address 0 (MMAL0 and MMAH0), etc.
BROADCAST_AND	4	0b	RW	Broadcast. Controls the inclusion of broadcast address filtering in the manageability filter decision (AND section).
VLAN_AND	12:5	0x0	RW	VLAN. Controls the inclusion of VLAN tag 0 to 7, respectively, in the manageability filter decision (AND section). Bit 5 corresponds to VLAN tag 0, etc.
IPV4_ADDRESS_AND	16:13	0x0	RW	IPv4 Address. Controls the inclusion of IPV4 address 0 to 3, respectively, in the manageability filter decision (AND section). Bit 13 corresponds to IPV4 address 0, etc. Note: These bits are also set for an ARP request packet if the target IP matches the IP address configured in the MIPAF register.
IPV6_ADDRESS_AND	20:17	0x0	RW	IPv6 Address. Controls the inclusion of IPV6 address 0 to 3, respectively, in the manageability filter decision (AND section). Bit 17 corresponds to IPV6 address 0, etc.
MAC_EXACT_OR	24:21	0x0	RW	Exact. Controls the inclusion of the exact MAC address 0 to 3 in the manageability filter decision (OR section). Bit 21 corresponds to the exact MAC address 0 (MMAL0 and MMAH0), etc.
BROADCAST_OR	25	0b	RW	Broadcast. Controls the inclusion of broadcast address filtering in the manageability filter decision (OR section).
MULTICAST_AND	26	0b	RW	Multicast. Controls the inclusion of multicast address filtering in the manageability filter decision (AND section). Broadcast packets are not included by this bit.
ARP_REQUEST_OR	27	0b	RW	ARP Request. Controls the inclusion of ARP request filtering in the manageability filter decision (OR section).
ARP_RESPONSE_OR	28	0b	RW	ARP Response. Controls the inclusion of ARP response filtering in the manageability filter decision (OR section).
NEIGHBOR_DISCOVERY_134_OR	29	0b	RW	Neighbor Discovery. Controls the inclusion of neighbor discovery filtering in the manageability filter decision (OR section). The neighbor type accepted by this filter is type 0x86 (134),
PORT_0X298_OR	30	0b	RW	Port 0x298. Controls the inclusion of port 0x298 filtering in the manageability filter decision (OR section).
PORT_0X26F_OR	31	0b	RW	Port 0x26F. Controls the inclusion of port 0x26F filtering in the manageability filter decision (OR section).

10.2.1.21.5 Management Only Traffic Register - PRT_MNG_MNGONLY[PRT] (0x00256A60 + 0x4*PRT, PRT=0...3)

The MNGONLY register enables exclusive filtering of certain types of traffic to the MC. Exclusive filtering enables the MC to define certain packets that are forwarded to the MC but not to the host. The packets are not forwarded to the host even if they pass the host L2 filtering process.



Each manageability decision filter (MDEF and MDEF_EXT) has a corresponding bit in the MNGONLY register. When a manageability decision filter (MDEF and MDEF_EXT) forwards a packet to manageability, it might also block the packet from being forwarded to the host if the corresponding MNGONLY bit is set.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
EXCLUSIVE_TO_MANAGEABILITY	7:0	0x0	RW	Exclusive to MNG. When set, indicates that packets forwarded by the manageability filters to manageability are not sent to the host. Bits 0...7 correspond to decision rules defined in registers MDEF[0...7] and MDEF_EXT[0...7].
Reserved	31:8	0x0	Reserved	Reserved.

10.2.1.21.6 Management Decision Filters VSI - PRT_MNG_MDEFVSI[n,PRT] (0x00256580 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

This register is used to define the VSIs that are used to receive packets that matched a specific MDEF. In case of a multiple match, the VSI assigned to the MDEF with the highest index is used.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MDEFVSI_2N	15:0	0x0	RW	Defines the VSI used for packets matching MDEF 2*n.
MDEFVSI_2NP1	31:16	0x0	RW	Defines the VSI used for packets matching MDEF 2*n+1.



10.2.1.21.7 Manageability MAC Address Low - PRT_MNG_MMAL[n,PRT] (0x00256480 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

These registers contain the lower bits of the 48-bit Ethernet address. The MMAL registers are written by internal firmware and are not accessible to the host for writing. The registers are used to filter manageability packets.

Reset — The MMAL registers are cleared on LAN_PWR_GOOD only. The initial values for this register can be loaded from the NVM after power-up reset.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MMAL	31:0	0x000000	RW	Manageability MAC Address Low. The lower 32 bits of the 48-bit Ethernet address. Note: Appears in big Endian order (LS byte of MMAL is first on the wire).

Note: The MMAL.MMAL field should be written in network order.

10.2.1.21.8 Manageability MAC Address High - PRT_MNG_MMAH[n,PRT] (0x00256380 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

These registers contain the upper bits of the 48-bit Ethernet address. The complete address is {MMAH,MMAL}. The MMAH registers are written by the MC and are not accessible to the host for writing. The registers are used to filter manageability packets. The initial values for this register can be loaded from the NVM after power-up reset or firmware reset.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MMAH	15:0	0x000000	RW	Manageability MAC Address High. The upper 16 bits of the 48 bit Ethernet address. Note: Appear in Big Endian order (MS byte of MMAH is last on the wire).
Reserved	31:16	0x0	Reserved	Reserved. Reads as 0x0. Ignored on write.

Note: The MMAH.MMAH field should be written in network order.



10.2.1.21.9 Management VLAN TAG Value - PRT_MNG_MAVTV[n,PRT] (0x00255900 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

The MAVTV registers are written by the MC and are not accessible to the host for writing. The registers are used to filter manageability packets as described in the Management section.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
VID	11:0	0x0	RW	Contains the VLAN ID that should be compared with the incoming packet inner VLAN ID if the corresponding bit in MDEF is set.
Reserved	31:12	0x0	Reserved	Reserved.

10.2.1.21.10 Management Ethernet Type Filters - PRT_MNG_METF[n,PRT] (0x00256780 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

The METF registers are written by the MC and are not accessible to the host for writing. The registers are used to filter manageability packets.

Reset — The METF registers are cleared on LAN_PWR_GOOD only. The initial values for this register might be loaded from the NVM after power-up reset.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
ETYPE	15:0	0x0	RW	EtherType value to be compared against the L2 EtherType field in the Rx packet. Note: Appears in little Endian order (high byte first on the wire).
Reserved	29:16	0x0	Reserved	Reserved
POLARITY	30	0b	RW	0b = Positive filter. Filter enters the decision filters if a match occurred. 1b = Negative filter. Filter enters the decision filters if a match did not occur.
Reserved	31	0b	Reserved	Reserved.



10.2.1.21.11 Manageability IPv6 Address Filter - PRT_MNG_MIPAF6[n,PRT] (0x00254200 + 0x20*n + 0x4*PRT, n=0...15, PRT=0...3)

The Manageability IPv6 Address Filter register stores IPv6 addresses for manageability filtering.

Note: These registers should be written in network order.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MIPAF	31:0	0x000000	RW	Manageability IP Address Filters. For each n, m, m=0...3, n=0...3, the MIPAF[m,n] register holds Dword 'n' of IPv6 filter 'm' (4 x IPv6 filters).

10.2.1.21.12 Manageability IPv4 Address Filter - PRT_MNG_MIPAF4[n,PRT] (0x00256280 + 0x20*n + 0x4*PRT, n=0...3, PRT=0...3)

The Manageability IPv4 Address Filter register stores IPv4 addresses for manageability filtering.

Note: These registers should be written in network order.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT.

Field	Bit(s)	Init.	Access Type	Description
MIPAF	31:0	0x000000	RW	Manageability IP Address Filters. For each n, m, m=0...3, n=0...3, the MIPAF[m,n] register holds Dword 'n' of IPv6 filter 'm' (4 x IPv6 filters).

10.2.1.21.13 Management Flex UDP/TCP Ports - PRT_MNG_MFUTP[n,PRT] (0x00254E00 + 0x20*n + 0x4*PRT, n=0...15, PRT=0...3)

Where each 32-bit register (n=0..15) refers to one UDP/TCP port filter.

The MFUTP registers are written by the MC and are not accessible to the host for writing. The registers are used to filter manageability packets.

Reset — The MFUTP registers are cleared on LAN_PWR_GOOD only. The initial values for this register can be loaded from the NVM after power-up reset.

Note: The MFUTP_N fields should be written in network order.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
MFUTP_N	15:0	0x0	RW	n-th Management Flex UDP/TCP Port.



Field	Bit(s)	Init.	Access Type	Description
UDP	16	0b	RW	Match if port is UDP.
TCP	17	0b	RW	Match if port is TCP.
SOURCE_DESTINATION	18	0b	RW	0b = Compare destination port. 1b = Compare source port.
Reserved	31:19	0x0	Reserved	Reserved.

10.2.1.21.14 Manageability Special Filters Modifiers - PRT_MNG_MSFM[PRT] (0x00256AA0 + 0x4*PRT, PRT=0...3)

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
PORT_26F_UDP	0	1b	RW	Port 0x26F match if protocol is UDP.
PORT_26F_TCP	1	1b	RW	Port 0x26F match if protocol is TCP.
PORT_298_UDP	2	1b	RW	Port 0x298 match if protocol is UDP.
PORT_298_TCP	3	1b	RW	Port 0x298 match if protocol is TCP.
IPV6_0_MASK	4	0b	RW	Compare only 24 LSB bits of IPv6 address 0 (MIPAF[0]).
IPV6_1_MASK	5	0b	RW	Compare only 24 LSB bits of IPv6 address 1 (MIPAF[1]).
IPV6_2_MASK	6	0b	RW	Compare only 24 LSB bits of IPv6 address 2 (MIPAF[2]).
IPV6_3_MASK	7	0b	RW	Compare only 24 LSB bits of IPv6 address 3 (MIPAF[3]).
Reserved	31:8	0x0	Reserved	Reserved.

10.2.1.21.15 Flexible TCO Filter Table Registers - Data - PRT_MNG_FTFT_DATA[n,PRT] (0x000852A0 + 0x20*n + 0x4*PRT, n=0...31, PRT=0...3)

The Flexible TCO Filter Table (FTFT) registers contain a 128-byte pattern and a corresponding 128-bit mask array.

If enabled, the first 128 bytes of the received packet are compared against the masked bytes in the PRT_MNG_FTFT_MASK registers.

The 128-byte filter is composed of 32 Dword entries, accompanied by a 128-bit mask (PRT_MNG_FTFT_MASK, one bit per filter byte). The bytes in each Dwords are written in network order. For example, byte0 written to bits [7:0], byte1 to bits [15:8] etc. The *Mask* field is set so that bit0 in the mask masks byte0, bit 1 masks byte 1 etc. A value of one in the *Mask* field means that the appropriate byte in the filter should be compared to the appropriate byte in the incoming packet.



The PRT_MNG_FTFT_LENGTH register contains a length field defining the number of bytes from the beginning of the packet compared by this filter. If actual packet length is less than length specified by this field, the filter fails. Otherwise, it depends on the result of actual byte comparison. The value should not be greater than 128.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
DWORD	31:0	0x000000	RW	Filter Data.

10.2.1.21.16 Flexible TCO Filter Table Registers - Mask - PRT_MNG_FTFT_MASK[n,PRT] (0x00085160 + 0x20*n + 0x4*PRT, n=0...7, PRT=0...3)

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT.

Field	Bit(s)	Init.	Access Type	Description
MASK	15:0	0x0	RW	Masks for the filter bytes. PRT_MNG_FTFT_MASK[0] = Mask for bytes 0:15. PRT_MNG_FTFT_MASK[1] = Mask for bytes 16:31. PRT_MNG_FTFT_MASK[6] = Mask for bytes 96:111. PRT_MNG_FTFT_MASK[7] = Mask for bytes. 112:127. 0b = Ignore. 1b = compare.
Reserved	31:16	0x0	Reserved	Reserved.

10.2.1.21.17 Flexible TCO Filter Table Registers - Length - PRT_MNG_FTFT_LENGTH[PRT] (0x00085260 + 0x4*PRT, PRT=0...3)

Contains the length of the filter.

Reset Source: POR

NVM Load: Firmware loaded

Scope: PRT

Field	Bit(s)	Init.	Access Type	Description
LENGTH	7:0	0x0	RW	This field contains the length of the filter defining the number of bytes from the beginning of the packet compared by this filter. If actual packet length is less than length specified by this field, the filter fails. Otherwise, it depends on the result of actual byte comparison. The value should not be greater than 128.
Reserved	31:8	0x0	Reserved	Reserved.



10.2.1.21.18 Hardware Arbitration Control - GL_MNG_HWARB_CTRL (0x000B6130)

Reset Source: POR

NVM Load: Type1/2 in POR register auto-load module

Scope: GL: No.

Field	Bit(s)	Init.	Access Type	Description
NCSI_ARB_EN	0	0b	RW	Hardware Arbitration Enable. If this bit is set, it is assumed the NCSI_ARB_IN and NCSI_ARB_OUT are connected to a hardware arbitration ring. Otherwise, the NCSI_ARB_IN pin is pulled up internally.
Reserved	31:1	0x0	Reserved	Reserved.

10.2.1.21.19 Firmware Status - GL_MNG_FWSM (0x000B6134)

This register reflects the firmware load status. All the bits in the registers are RW bits and the firmware implements the functionality. Bits 15:0 are reset by EMP reset.

Reset Source: POR

NVM Load: No

Scope: GL

Field	Bit(s)	Init.	Access Type	Description
FW_MODES ¹	2:0	0x0	RW	Firmware Mode. Indicates firmware mode of operation. 000b = Normal mode. 001b = Debug mode. 010b = GLOBR/CORER recovery mode. 011b = Debug and GLOBR/CORER Recovery mode. 100b = NVM recovery mode. 101b = Debug and NVM recovery mode.
Reserved	9:3	0x0	Reserved	Reserved.
EEP_RELOAD_IND	10	0b	RW	NVM Reload Indication. Set to 1b after firmware reloads the NVM configuration after a Core reset Cleared by firmware once the first AQ command is received from one of the drivers.
CRC_ERROR_MODULE	14:11	0x0	RW	Index of (first) module for which a CRC error was found by EMP check. 0x0 = No CRC error found by EMP. 0x1 = CRC error on EMP Image module. 0x2 = CRC error on PCIe analog module. 0x3 = CRC error on PHY analog module. 0x4 = CRC error on EMP global module. 0x5 = CRC error on manageability module. 0x6 = CRC error on EMP settings module. 0x7 = CRC error on external PHY module.
FW_STATUS_VALID	15	0b	RW	Firmware Valid Bit. Hardware clears bits [15:0] in EMP reset de-assertion so software knows firmware status is invalid. Firmware should set this bit to 1b when it is ready (end of init sequence). Whenever setting this bit to 1b, a PFINT_ICR0.ADMINQ interrupt shall be issued to host.



Field	Bit(s)	Init.	Access Type	Description
RESET_CNT	18:16	0x0	RW	Reset Counter. Firmware increments the count on every EMP reset. After seven EMP reset events, the counter remains stuck at seven and does not wrap around.
EXT_ERR_IND	24:19	0x0	RW	External Error Indication. Firmware writes here the reason that firmware operation has stopped. For example, NVM CRC error, etc. Possible values: 0x0 = No error. 0x1 = CRC error on a module handled by EMP. Refer to CRC_ERROR_MODULE field for details. 0x2 = Switch module failed. 0x3 = Scheduler module failed. 0x4 = DCB module failed. 0x5 = Link module failed. 0x6 = LLDP module failed. 0x7 = Manage module failed. 0x8 - 0x39 = Reserved. Note: Following error detection and GL_MNG_FWSM.EXT_IND_ERR update, the PFINT_ICR0.ADMINQ bit is set and an interrupt is sent to the host. However when values of 0x0 are placed in this field, the PFINT_ICR0.ADMINQ bit is not set and an interrupt is not generated.
Reserved	25	0b	Reserved	Reserved.
PHY_SERDES0_CONFIG_ERR	26	0b	RW	PHY/SerDes Configuration Error Indication (Port 0). Set by firmware when it fails to configure LAN PHY/SerDes. Cleared by firmware upon successful configuration of LAN PHY/SerDes.
PHY_SERDES1_CONFIG_ERR	27	0b	RW	PHY/SerDes Configuration Error Indication (Port 1). Set by firmware when it fails to configure LAN PHY/SerDes. Cleared by firmware upon successful configuration of LAN PHY/SerDes.
PHY_SERDES2_CONFIG_ERR	28	0b	RW	PHY/SerDes Configuration Error Indication (Port 2). Set by firmware when it fails to configure LAN PHY/SerDes. Cleared by firmware upon successful configuration of LAN PHY/SerDes.
PHY_SERDES3_CONFIG_ERR	29	0b	RW	PHY/SerDes Configuration Error Indication (Port 3). Set by firmware when it fails to configure LAN PHY/SerDes. Cleared by firmware upon successful configuration of LAN PHY/SerDes.
Reserved	31:30	0x0	Reserved	Reserved.

1. This register is used for firmware debug purpose only.

10.2.2 BAR3 Registers Summary

Table 10-8. BAR3 Registers Summary

Offset / Alias Offset	Abbreviation	Name	Section
PF - MSI-X Table Registers			
0x00000000 + 0x10*n, n=0...128	MSIX_TADD[n]	MSI-X Message Address Low	Section 10.2.3.1.1
0x00000004 + 0x10*n, n=0...128	MSIX_TUADD[n]	MSI-X Message Address High	Section 10.2.3.1.2
0x00000008 + 0x10*n, n=0...128	MSIX_TMSG[n]	MSI-X Message Data	Section 10.2.3.1.3



Table 10-8. BAR3 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x0000000C + 0x10*n, n=0...128	MSIX_TVCTRL[n]	MSI-X Vector Control	Section 10.2.3.1.4
0x00001000 + 0x4*n, n=0...5	MSIX_PBA[n]	MSI-X PBA Structure	Section 10.2.3.1.5
0x00002000 + 0x4*n, n=0...19	VFMSIX_PBA[n]	VF MSI-X PBA Structure	Section 10.2.3.1.6
0x00002100 + 0x10*n, n=0...639	VFMSIX_TADD[n]	VF MSI-X Message Address Low	Section 10.2.3.1.7
0x00002104 + 0x10*n, n=0...639	VFMSIX_TUADD[n]	VF MSI-X Message Address High	Section 10.2.3.1.8
0x00002108 + 0x10*n, n=0...639	VFMSIX_TMSG[n]	VF MSI-X Message Data	Section 10.2.3.1.9
0x0000210C + 0x10*n, n=0...639	VFMSIX_TVCTRL[n]	VF MSI-X Vector Control	Section 10.2.3.1.10

10.2.3 Detailed Register Description - PF BAR3

10.2.3.1 PF - MSI-X Table Registers

This category contains registers in the separate MSI-X BAR.

10.2.3.1.1 MSI-X Message Address Low - MSIX_TADD[n] (0x00000000 + 0x10*n, n=0...128; RW)

Message lower address for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTADD10	1:0	00b	RW	Message Address 1:0 For proper Dword alignment, software must always write zeros to these two bits. Otherwise, the result is undefined. The state of these bits after reset must be 0b. These bits are permitted to be read-only or read/write.
MSIXTADD	31:2	0x0	RW	Message Address System-specified message lower address. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the lower portion of the Dword-aligned address (AD[31:02]) for the memory write transaction. This field is read/write.



10.2.3.1.2 MSI-X Message Address High - MSIX_TUADD[n] (0x00000004 + 0x10*n, n=0...128; RW)

Message upper address for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTUADD	31:0	0x0	RW	Message Upper Address System-specified message upper address bits. If this field is zero, Single Address Cycle (SAC) messages are used. If this field is non-zero, Dual Address Cycle (DAC) messages are used. This field is read/write.

10.2.3.1.3 MSI-X Message Data - MSIX_TMSG[n] (0x00000008 + 0x10*n, n=0...128; RW)

Message data for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTMSG	31:0	0x0	RW	Message Data System-specified message data. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the data driven on AD[31:0] during the memory write transaction's data phase. This field is read/write.

10.2.3.1.4 MSI-X Vector Control - MSIX_TVCTRL[n] (0x0000000C + 0x10*n, n=0...128; RW)

Vector control for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MASK	0	1b	RW	Mask Bit 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 are still capable of sending an equivalent message unless they are also masked. This bit's state after reset is 1b (entry is masked).
RESERVED	31:1	0x0	RSV	Reserved. After reset, the state of these bits must be 0b. However, for potential future use, software must preserve the value of these reserved bits when modifying the value of other Vector Control bits. If software modifies the value of these reserved bits, the result is undefined.



10.2.3.1.5 MSI-X PBA Structure - MSIX_PBA[n] (0x00001000 + 0x4*n, n=0...5; RO)

Pending bits for MSI-X PBA entries.

Field	Bit(s)	Init.	Type	Description
PENBIT	31:0	0x0	RO	MSI-X Pending Bits Each bit is set to 1b when the appropriate interrupt request is set and cleared to 0b when the appropriate interrupt request is cleared.

10.2.3.1.6 VF MSI-X PBA Structure - VFMSIX_PBA[n] (0x00002000 + 0x4*n, n=0...19; RO)

Pending bits for MSI-X PBA entries.

Field	Bit(s)	Init.	Type	Description
PENBIT	31:0	0x0	RO	MSI-X Pending Bits Each bit is set to 1b when the appropriate interrupt request is set and cleared to 0b when the appropriate interrupt request is cleared.

10.2.3.1.7 VF MSI-X Message Address Low - VFMSIX_TADD[n] (0x00002100 + 0x10*n, n=0...639; RW)

Message address for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTADD10	1:0	00b	RW	Message Address 1:0 For proper Dword alignment, software must always write zeros to these two bits. Otherwise, the result is undefined. The state of these bits after reset must be 0b. These bits are permitted to be read-only or read/write.
MSIXTADD	31:2	0x0	RW	Message Address System-specified message lower address. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the lower portion of the Dword-aligned address (AD[31:02]) for the memory write transaction. This field is read/write.



10.2.3.1.8 VF MSI-X Message Address High - VFMSIX_TUADD[n] (0x00002104 + 0x10*n, n=0...639; RW)

Message upper address for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTUADD	31:0	0x0	RW	Message Upper Address System-specified message upper address bits. If this field is zero, Single Address Cycle (SAC) messages are used. If this field is non-zero, Dual Address Cycle (DAC) messages are used. This field is read/write.

10.2.3.1.9 VF MSI-X Message Data - VFMSIX_TMSG[n] (0x00002108 + 0x10*n, n=0...639; RW)

Message data for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTMSG	31:0	0x0	RW	Message Data System-specified message data. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the data driven on AD[31:0] during the memory write transaction's data phase. This field is read/write.

10.2.3.1.10 VF MSI-X Vector Control - VFMSIX_TVCTRL[n] (0x0000210C + 0x10*n, n=0...639) msix_regs

Vector control for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MASK	0	1b	RW	Mask Bit 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 are still capable of sending an equivalent message unless they are also masked. This bit's state after reset is 1b (entry is masked).
RESERVED	31:1	0x0	RSV	Reserved. After reset, the state of these bits must be 0b. However, for potential future use, software must preserve the value of these reserved bits when modifying the value of other Vector Control bits. If software modifies the value of these reserved bits, the result is undefined.



10.3 Device Registers - VF

10.3.1 VF Registers Mapping in the PF Space

Table 10-9. VF Registers Summary

Abbreviation	Virtual Address	Physical Address
VFMSIX_TADD	0x00000000 + 0x10*n, n=0...16	0x00002100 + 0x10*n, n=0...639
QTX_TAIL	0x00000000 + 0x4*Q, Q=0...15	0x00108000 + 0x4*Q, Q=0...1535
VFMSIX_TUADD	0x00000004 + 0x10*n, n=0...16	0x00002104 + 0x10*n, n=0...639
VFMSIX_TMSG	0x00000008 + 0x10*n, n=0...16	0x00002108 + 0x10*n, n=0...639
VFMSIX_TVCTRL	0x0000000C + 0x10*n, n=0...16	0x0000210C + 0x10*n, n=0...639
VFMSIX_PBA	0x00002000	0x00002000 + 0x4*n, n=0...19
QRX_TAIL	0x00002000 + 0x4*Q, Q=0...15	0x00128000 + 0x4*Q, Q=0...1535
VFINT_ITRN	0x00002800 + 0x40*n + 0x4*INTVF, n=0...2, INTVF=0...15	0x00020000 + 0x800*n + 0x4*INTVF, n=0...2, INTVF=0...511
VFINT_DYN_CTLN	0x00003800 + 0x4*INTVF, INTVF=0...15	0x00024800 + 0x4*INTVF, INTVF=0...511
VFINT_ICR0	0x00004800	0x0002BC00 + 0x4*VF, VF=0...127
VFINT_ITR0	0x00004C00 + 0x4*n, n=0...2	0x00028000 + 0x400*n + 0x4*VF, n=0...2, VF=0...127
VFINT_ICR0_ENA	0x00005000	0x0002C000 + 0x4*VF, VF=0...127
VFINT_STAT_CTL0	0x00005400	0x0002A000 + 0x4*VF, VF=0...127
VFINT_DYN_CTL0	0x00005C00	0x0002A400 + 0x4*VF, VF=0...127
VF_ARQBAH	0x00006000	0x00081400 + 0x4*VF, VF=0...127
VF_ATQH	0x00006400	0x00082000 + 0x4*VF, VF=0...127
VF_ATQLEN	0x00006800	0x00081800 + 0x4*VF, VF=0...127
VF_ARQBAL	0x00006C00	0x00080C00 + 0x4*VF, VF=0...127
VF_ARQT	0x00007000	0x00082C00 + 0x4*VF, VF=0...127
VF_ARQH	0x00007400	0x00082400 + 0x4*VF, VF=0...127
VF_ATQBAH	0x00007800	0x00081000 + 0x4*VF, VF=0...127
VF_ATQBAL	0x00007C00	0x00080800 + 0x4*VF, VF=0...127
VF_ARQLEN	0x00008000	0x00081C00 + 0x4*VF, VF=0...127
VF_ATQT	0x00008400	0x00082800 + 0x4*VF, VF=0...127
VFGEN_RSTAT	0x00008800	0x00074400 + 0x4*VF, VF=0...127
VFQF_HENA	0x0000C400 + 0x4*n, n=0...1	0x00230800 + 0x400*n + 0x4*VF, n=0...1, VF=0...127
VFQF_HKEY	0x0000CC00 + 0x4*n, n=0...12	0x00228000 + 0x400*n + 0x4*VF, n=0...12, VF=0...127
VFQF_HLUT	0x0000D000 + 0x4*n, n=0...15	0x00220000 + 0x400*n + 0x4*VF, n=0...15, VF=0...127
VFQF_HREGION	0x0000D400 + 0x4*n, n=0...7	0x0022E000 + 0x400*n + 0x4*VF, n=0...7, VF=0...127
PFPCI_VF_FLUSH_DONE	0x0000E400	0x0009C600 + 0x4*VF, VF=0...127



10.3.2 BAR0 Registers Summary

Table 10-10. BAR0 Registers Summary

Offset / Alias Offset	Abbreviation	Name	Section
PF - PCIe Registers			
0x0000E400	PFPCI_VF_FLUSH_DONE	PCIe VF Flush Done	Section 10.3.3.1.1
VF - General Registers			
0x00008800	VFGEN_RSTAT	VF Reset Status	Section 10.3.3.2.1
VF - Interrupts			
0x00002800 + 0x40*n + 0x4*INTVF, n=0...2, INTVF=0...15	VFINT_ITRN[n,INTVF]	VF Interrupt Throttling for Interrupt N	Section 10.3.3.3.1
0x00003800 + 0x4*INTVF, INTVF=0...15	VFINT_DYN_CTLN[INTVF]	VF Interrupt N Dynamic Control	Section 10.3.3.3.2
0x00004800	VFINT_ICR0	VF Interrupt Zero Cause	Section 10.3.3.3.3
0x00004C00 + 0x4*n, n=0...2	VFINT_ITR0[n]	VF Interrupt Throttling for Interrupt Zero	Section 10.3.3.3.4
0x00005000	VFINT_ICR0_ENA	VF Interrupt Zero Cause Enablement	Section 10.3.3.3.5
0x00005400	VFINT_STAT_CTL0	VF Interrupt Zero Static Control	Section 10.3.3.3.6
0x00005C00	VFINT_DYN_CTL0	VF Interrupt Zero Dynamic Control	Section 10.3.3.3.7
VF - Admin Queue			
0x00006000	VF_ARQBAH	VF Admin Receive Queue Base Address High	Section 10.3.3.4.1
0x00006400	VF_ATQH	VF Admin Transmit Head	Section 10.3.3.4.2
0x00006800	VF_ATQLEN	VF Admin Transmit Queue Length	Section 10.3.3.4.3
0x00006C00	VF_ARQBAL	VF Admin Receive Queue Base Address Low	Section 10.3.3.4.4
0x00007000	VF_ARQT	VF Admin Receive Queue Tail	Section 10.3.3.4.5
0x00007400	VF_ARQH	VF Admin Receive Queue Head	Section 10.3.3.4.6
0x00007800	VF_ATQBAH	VF Admin Transmit Queue Base Address High	Section 10.3.3.4.7
0x00007C00	VF_ATQBAL	VF Admin Transmit Queue Base Address Low	Section 10.3.3.4.8
0x00008000	VF_ARQLEN	VF Admin Receive Queue Length	Section 10.3.3.4.9
0x00008400	VF_ATQT	VF Admin Transmit Tail	Section 10.3.3.4.10
VF - LAN Transmit Receive Registers			
0x00000000 + 0x4*Q, Q=0...15	QTX_TAIL[Q]	Global Transmit Queue Tail	Section 10.3.3.5.1
0x00002000 + 0x4*Q, Q=0...15	QRX_TAIL[Q]	Global Receive Queue Tail	Section 10.3.3.5.2
VF - Rx Filters Registers			



Table 10-10. BAR0 Registers Summary (Continued)

Offset / Alias Offset	Abbreviation	Name	Section
0x0000C400 + 0x4*n, n=0...1	VFQF_HENA[n]	VF Queue Filter Hash Enabled Packet Type	Section 10.3.3.6.1
0x0000CC00 + 0x4*n, n=0...12	VFQF_HKEY[n]	VF Queue Filter Hash Key	Section 10.3.3.6.2
0x0000D000 + 0x4*n, n=0...15	VFQF_HLUT[n]	VF Queue Filter Hash LUT	Section 10.3.3.6.3
0x0000D400 + 0x4*n, n=0...7	VFQF_HREGION[n]	VF Queue Filter Hash Region of Queues	Section 10.3.3.6.4

10.3.3 Detailed Register Description - VF BAR0

10.3.3.1 PF - PCIe Registers

10.3.3.1.1 PCIe VF Flush Done - PFPCI_VF_FLUSH_DONE (0x0000E400; RO)

Field	Bit(s)	Init.	Type	Description
FLUSH_DONE	0	0b	RO	Flush Done.
RESERVED	31:1	0x0	RSV	Reserved.

10.3.3.2 VF - General Registers

This section describes the registers allocated to a VF for generic control and status. These registers are tied to the VF and are not dependent of any resource allocation.

10.3.3.2.1 VF Reset Status - VFGEN_RSTAT (0x00008800; RW)

Field	Bit(s)	Init.	Type	Description
VFR_STATE	1:0	00b	RW	VFR State Defines the VFR reset progress as follows: 00b = VFR in progress 01b = VFR completed 10b = Reserved 11b = Reserved This field is used to communicate the reset progress to the VF with no impact on hardware functionality.
RESERVED	31:2	0x0	RSV	Reserved.



10.3.3.3 VF - Interrupts

10.3.3.3.1 VF Interrupt Throttling for Interrupt N - VFINT_ITRN[n,INTVF] (0x00002800 + 0x40*n + 0x4*INTVF, n=0...2, INTVF=0...15; RW)

Register index 'n' relates to interrupt 'n+1', while interrupt zero is controlled by the VFINT_ITRO register (Section 10.2.2.10.5).

Field	Bit(s)	Init.	Type	Description
INTERVAL	11:0	0x0	RW	Interval ITR 'n' interval, where 'n' is the register index = 0,1,2 for the three ITRs per interrupt. It is defined in 2 μ s units, enabling interval range from zero to 8160 μ s (0xFF0). Setting the <i>INTERVAL</i> to zero enables immediate interrupt. This register can also be programmed by setting the <i>INTERVAL</i> field in the matched xxINT_DYN_CTLx register.
RESERVED	31:12	0x0	RSV	Reserved.

10.3.3.3.2 VF Interrupt N Dynamic Control - VFINT_DYN_CTLN[INTVF] (0x00003800 + 0x4*INTVF, INTVF=0...15; RW)

Register index 'n' relates to interrupt 'n+1', while interrupt zero is controlled by the VFINT_DYN_CTL0 register (Section 10.2.2.10.7).

Field	Bit(s)	Init.	Type	Description
INTENA	0	0b	RW	Interrupt Enable 0 = Interrupt disabled. 1 = Interrupt enabled. Refer to auto-clear policy in the "Interrupt Enablement" section. This bit is meaningful only if the INTENA_MSK flag in this register is not set.
CLEARPBA	1	0b	RW1C	Clear PBA Setting this bit clears the matched PBA bit. This bit is auto-cleared by hardware.
SWINT_TRIG	2	0b	RW1C	Software Interrupt Trigger When this bit is set, a software interrupt is triggered. This bit is auto-cleared by hardware.
ITR_INDX	4:3	00b	RW1C	ITR Index Defines the ITR index to be updated, as follows: 00b = ITR0 01b = ITR1 10b = ITR2 11b = No ITR Update This field is auto-cleared by hardware.
INTERVAL	16:5	0x0	RW1C	Interval The interval for the ITR defined by the ITR_INDX field in this register. This field is auto-cleared by hardware.
RESERVED	23:17	0x0	RSV	Reserved.



Field	Bit(s)	Init.	Type	Description
SW_ITR_IND_X_ENA	24	0b	RW1C	Software ITR Index Enable Enables the programming of the SW_ITR_IND_X field in this register. This flag is auto-cleared by hardware.
SW_ITR_IND_X	26:25	00b	RW	Software ITR Index ITR index of the software interrupt: 00b = ITR0 01b = ITR1 10b = ITR2 11b = NoITR When programming this field, the SW_ITR_IND_X_ENA flag in this register should be set as well.
RESERVED	30:27	0x0	RSV	Reserved.
INTENA_MSK	31	0b	RW1C	Interrupt Enable Mask When this bit is set, the INTENA setting does not impact the device setting. This bit is auto-cleared by hardware.

10.3.3.3.3 VF Interrupt Zero Cause - VFINT_ICR0 (0x00004800; RCW)

Field	Bit(s)	Init.	Type	Description
INTEVENT	0	0b	RCW	Interrupt Event indication This bit is set on assertion of any causes for this interrupt, and cleared when the interrupt is asserted to the Rate Limit logic.
RESERVED	29:1	0x0	RSV	Reserved.
ADMINQ	30	0b	RCW	Admin Queue Send/Receive Admin queue interrupt indication.
SWINT	31	0b	RCW	Software Interrupt indication

10.3.3.3.4 VF Interrupt Throttling for Interrupt Zero - VFINT_ITR0[n] (0x00004C00 + 0x4*n, n=0...2; RW)

Field	Bit(s)	Init.	Type	Description
INTERVAL	11:0	0x0	RW	Interval ITR 'n' interval, where 'n' is the register index = 0,1,2 for the three ITRs per interrupt. It is defined in 2 μs units, enabling interval range from zero to 8160 μs (0xFF0). Setting the INTERVAL to zero enables immediate interrupt. This register can also be programmed by setting the INTERVAL field in the matched xxINT_DYN_CTLx register.
RESERVED	31:12	0x0	RSV	Reserved.

10.3.3.3.5 VF Interrupt Zero Cause Enablement - VFINT_ICR0_ENA (0x00005000; RW)

Field	Bit(s)	Init.	Type	Description
RESERVED	29:0	0x0	RSV	Reserved.



Field	Bit(s)	Init.	Type	Description
ADMINQ	30	0b	RW	Admin Queue Enable this interrupt at 1b.
RESERVED	31	0b	RW	Reserved.

10.3.3.3.6 VF Interrupt Zero Static Control - VFINT_STAT_CTL0 (0x00005400; RW)

In case of MSI or Legacy INTA mode of operation, interrupt zero is the only valid interrupt.

Field	Bit(s)	Init.	Type	Description
RESERVED	1:0	00b	RSV	Reserved.
OTHER_ITR_INDX	3:2	00b	RW	Other ITR Index ITR index of the "other" interrupt causes: 00b = ITR0 01b = ITR1 10b = ITR2 11b = NoITR
RESERVED	31:4	0x0	RSV	Reserved.

10.3.3.3.7 VF Interrupt Zero Dynamic Control - VFINT_DYN_CTL0 (0x00005C00; RW)

Field	Bit(s)	Init.	Type	Description
INTENA	0	0b	RW	Interrupt Enable 0 = Interrupt disabled. 1 = Interrupt enabled. Refer to auto-clear policy in the "Interrupt Enablement" section. This bit is meaningful only if the INTENA_MSK flag in this register is not set.
CLEARPBA	1	0b	RW1C	Clear PBA Setting this bit clears the matched PBA bit. This bit is auto-cleared by hardware.
SWINT_TRIG	2	0b	RW1C	Software Interrupt Trigger When this bit is set, a software interrupt is triggered. This bit is auto-cleared by hardware.
ITR_INDX	4:3	00b	RW1C	ITR Index Defines the ITR index to be updated, as follows: 00b = ITR0 01b = ITR1 10b = ITR2 11b = No ITR Update This field is auto-cleared by hardware.
INTERVAL	16:5	0x0	RW1C	Interval The interval for the ITR defined by the ITR_INDX field in this register. This field is auto-cleared by hardware.
RESERVED	23:17	0x0	RSV	Reserved.



Field	Bit(s)	Init.	Type	Description
SW_ITR_INDX_ENA	24	0b	RW1C	Software ITR Index Enable Enables the programming of the SW_ITR_INDX field in this register. This flag is auto-cleared by hardware.
SW_ITR_INDX	26:25	00b	RW	Software ITR Index ITR index of the software interrupt: 00b = ITR0 01b = ITR1 10b = ITR2 11b = NoITR When programming this field, the SW_ITR_INDX_ENA flag in this register should be set as well.
RESERVED	30:27	0x0	RSV	Reserved.
INTENA_MSK	31	0b	RW1C	Interrupt Enable Mask When this bit is set, the INTENA setting does not impact the device setting. This bit is auto-cleared by hardware.

10.3.3.4 VF - Admin Queue

10.3.3.4.1 VF Admin Receive Queue Base Address High - VF_ARQBAH (0x00006000; RW)

Field	Bit(s)	Init.	Type	Description
ARQBAH	31:0	0x0	RW	Admin Receive Queue Base Address High Receive descriptor base address high.

10.3.3.4.2 VF Admin Transmit Head - VF_ATQH (0x00006400; RW)

Field	Bit(s)	Init.	Type	Description
ATQH	9:0	0x0	RW	Admin Transmit Queue Head Transmit queue head pointer. At queue initialization the software clears the head pointer and during nominal operation the Firmware increments the head following command execution.
RESERVED	31:10	0x0	RSV	Reserved.

10.3.3.4.3 VF Admin Transmit Queue Length - VF_ATQLEN (0x00006800; RW)

Field	Bit(s)	Init.	Type	Description
ATQLEN	9:0	0x0	RW	Admin Transmit Queue Length Descriptor ring length. Max size is 1024.
RESERVED	27:10	0x0	RSV	Reserved.



Field	Bit(s)	Init.	Type	Description
ATQVFE	28	0b	RW	Admin Transmit Queue VF Error Set by firmware on a PF queue when one of its VFs has an admin queue error.
ATQOVFL	29	0b	RW	Admin Transmit Queue Overflow Error Set by firmware when a message is lost because there is no room in the queue.
ATQCRIT	30	0b	RW	Admin Transmit Queue Critical Error Set by firmware when a critical error is detected on this queue.
ATQENABLE	31	0b	RW	Admin Transmit Queue Enable Set by driver to indicate that the queue is active. When setting the enable bit, software should initialize all other fields. This flag is cleared by VFR.

10.3.3.4.4 VF Admin Receive Queue Base Address Low - VF_ARQBAL (0x00006C00; RW)

Field	Bit(s)	Init.	Type	Description
ARQBAL	31:0	0x0	RW	Admin Receive Queue Base Address Low Receive descriptor base address low. Must be 64-byte aligned.

10.3.3.4.5 VF Admin Receive Queue Tail - VF_ARQT (0x00007000; RW)

Field	Bit(s)	Init.	Type	Description
ARQT	9:0	0x0	RW	Admin Receive Queue Tail Receive queue tail pointer. Incremented to indicate that there are new valid descriptors on the ring. Software may only write to this register once the queue is fully configured, and clears to zero at queue initialization.
RESERVED	31:10	0x0	RSV	Reserved.

10.3.3.4.6 VF Admin Receive Queue Head - VF_ARQH (0x00007400; RW)

Field	Bit(s)	Init.	Type	Description
ARQH	9:0	0x0	RW	Admin Receive Queue Head Transmit queue head pointer. At queue initialization, the software clears the head pointer. During nominal operation, the firmware increments the head following command execution.
RESERVED	31:10	0x0	RSV	Reserved.

10.3.3.4.7 VF Admin Transmit Queue Base Address High - VF_ATQBAH (0x00007800; RW)

Field	Bit(s)	Init.	Type	Description
ATQBAH	31:0	0x0	RW	Admin Transmit Queue Base Address High Transmit descriptor base address high.



10.3.3.4.8 VF Admin Transmit Queue Base Address Low - VF_ATQBAL (0x00007C00; RW)

Field	Bit(s)	Init.	Type	Description
ATQBAL	31:0	0x0	RW	Admin Transmit Queue Base Address Low Transmit descriptor base address low. Must be 64-byte aligned.

10.3.3.4.9 VF Admin Receive Queue Length - VF_ARQLEN (0x00008000; RW)

Field	Bit(s)	Init.	Type	Description
ARQLEN	9:0	0x0	RW	Admin Receive Queue Length Descriptor ring length. Max size is 1024.
RESERVED	27:10	0x0	RSV	Reserved.
ARQVFE	28	0b	RW	Admin Receive Queue VF Error Set by firmware on a PF queue when one of its VFs has an admin queue error.
ARQOVFL	29	0b	RW	Admin Receive Queue Overflow Error Set by firmware when a message is lost because there is no room in the queue.
ARQCRIT	30	0b	RW	Admin Receive Queue Critical Error Set by firmware when a critical error is detected on this queue.
ARQENABLE	31	0b	RW	Admin Receive Queue Enable Set by driver to indicate that the queue is active. When setting the enable bit, software should initialize all other fields. This flag is cleared by PFR.

10.3.3.4.10 VF Admin Transmit Tail - VF_ATQT (0x00008400; RW)

Field	Bit(s)	Init.	Type	Description
ATQT	9:0	0x0	RW	Admin Transmit Queue Tail Transmit queue tail pointer. Incremented to indicate that there are new valid descriptors on the ring. Software may only write to this register once both transmit and receive queues are properly initialized, and clears to zero at queue initialization.
RESERVED	31:10	0x0	RSV	Reserved.



10.3.3.5 VF - LAN Transmit Receive Registers

10.3.3.5.1 Global Transmit Queue Tail - QTX_TAIL[Q] (0x00000000 + 0x4*Q, Q=0...15; RW)

Field	Bit(s)	Init.	Type	Description
TAIL	12:0	0x0	RW	Transmit Tail Defines the first descriptor that the software prepares for the hardware (it is the last valid descriptor plus one). The Tail is a relative descriptor index to the beginning of the transmit descriptor ring.
RESERVED	31:13	0x0	RSV	Reserved.

10.3.3.5.2 Global Receive Queue Tail - QRX_TAIL[Q] (0x00002000 + 0x4*Q, Q=0...15; RW)

Field	Bit(s)	Init.	Type	Description
TAIL	12:0	0x0	RW	Receive Tail Defines the first descriptor that the software hands to the hardware (it is the last valid descriptor plus one). The Tail is a relative descriptor index to the beginning of the receive descriptor ring.
RESERVED	31:13	0x0	RSV	Reserved.

10.3.3.6 VF - Rx Filters Registers

10.3.3.6.1 VF Queue Filter Hash Enabled Packet Type - VFQF_HENA[n] (0x0000C400 + 0x4*n, n=0...1; RW)

Accessing this register should be done by the Receive Control CSR Read / Write Admin command rather than direct access.

Field	Bit(s)	Init.	Type	Description
PTYPE_ENA	31:0	0x0	RW	Packet Type Enable Packet Type Enablement of the Hash filter for the function. Bit 'm' in register 'n' enables packet type '32*n+m' as defined in the "Packet Types for the Classification Filters" table.

10.3.3.6.2 VF Queue Filter Hash Key - VFQF_HKEY[n] (0x0000CC00 + 0x4*n, n=0...12; RW)

Field	Bit(s)	Init.	Type	Description
KEY_0	7:0	0x0	RW	Key 0 Toeplitz key byte '4*n+0', where 'n' is the register index.



Field	Bit(s)	Init.	Type	Description
KEY_1	15:8	0x0	RW	Key 1 Toeplitz key byte '4*n+1', where 'n' is the register index.
KEY_2	23:16	0x0	RW	Key 2 Toeplitz key byte '4*n+2', where 'n' is the register index.
KEY_3	31:24	0x0	RW	Key 3 Toeplitz key byte '4*n+3', where 'n' is the register index.

10.3.3.6.3 VF Queue Filter Hash LUT - VFQF_HLUT[n] (0x0000D000 + 0x4*n, n=0...15; RW)

Field	Bit(s)	Init.	Type	Description
LUT0	3:0	0x0	RW	LUT 0 Hash redirection LUT entry '4*n', where 'n' is the register index.
RESERVED	7:4	0x0	RSV	Reserved.
LUT1	11:8	0x0	RW	LUT 1 Hash redirection LUT entry '4*n+1', where 'n' is the register index.
RESERVED	15:12	0x0	RSV	Reserved.
LUT2	19:16	0x0	RW	LUT 2 Hash redirection LUT entry '4*n+2', where 'n' is the register index.
RESERVED	23:20	0x0	RSV	Reserved.
LUT3	27:24	0x0	RW	LUT 3 Hash redirection LUT entry '4*n+3', where 'n' is the register index.
RESERVED	31:28	0x0	RSV	Reserved.

10.3.3.6.4 VF Queue Filter Hash Region of Queues - VFQF_HREGION[n] (0x0000D400 + 0x4*n, n=0...7; RW)

Accessing this register should be done by the Receive Control CSR Read / Write Admin command rather than direct access.

Field	Bit(s)	Init.	Type	Description
OVERRIDE_ENA_0	0	0b	RW	Override Enable 0 Override Traffic Class Region for packet type '8*n', where 'n' is the register index.
REGION_0	3:1	000b	RW	Region 0 Receive queue region for packet type '8*n', where 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_1	4	0b	RW	Override Enable 1 Override Traffic Class Region for packet type '8*n+1', where 'n' is the register index.
REGION_1	7:5	000b	RW	Region 1 Receive queue region for packet type '8*n+1', where 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.



Field	Bit(s)	Init.	Type	Description
OVERRIDE_ENA_2	8	0b	RW	Override Enable 2 Override Traffic Class Region for packet type '8*n+2', where 'n' is the register index.
REGION_2	11:9	000b	RW	Region 2 Receive queue region for packet type '8*n+2', where 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_3	12	0b	RW	Override Enable 3 Override Traffic Class Region for packet type '8*n+3', where 'n' is the register index.
REGION_3	15:13	000b	RW	Region 3 Receive queue region for packet type '8*n+3', where 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_4	16	0b	RW	Override Enable 4 Override Traffic Class Region for packet type '8*n+4', where 'n' is the register index.
REGION_4	19:17	000b	RW	Region 4 Receive queue region for packet type '8*n+4', where 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_5	20	0b	RW	Override Enable 5 Override Traffic Class Region for packet type '8*n+5', where 'n' is the register index.
REGION_5	23:21	000b	RW	Region 5 Receive queue region for packet type '8*n+5', where 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_6	24	0b	RW	Override Enable 6 Override Traffic Class Region for packet type '8*n+6', where 'n' is the register index.
REGION_6	27:25	000b	RW	Region 6 Receive queue region for packet type '8*n+6', where 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.
OVERRIDE_ENA_7	28	0b	RW	Override Enable 7 Override Traffic Class Region for packet type '8*n+7', where 'n' is the register index.
REGION_7	31:29	000b	RW	Region 7 Receive queue region for packet type '8*n+7', where 'n' is the register index. This field is meaningful only if the OVERRIDE_ENA_1 flag is set.



10.3.4 BAR3 Registers Summary

Table 10-11. BAR3 Registers Summary

Offset / Alias Offset	Abbreviation	Name	Block	Section
VF - MSI-X Table Registers				
0x00000000 + 0x10*n, n=0...16	VFMSIX_TADD[n]	VF MSI-X Message Address Low	VF_BAR	Section 10.3.5.1.1
0x00000004 + 0x10*n, n=0...16	VFMSIX_TUADD[n]	VF MSI-X Message Address High	VF_BAR	Section 10.3.5.1.2
0x00000008 + 0x10*n, n=0...16	VFMSIX_TMSG[n]	VF MSI-X Message Data	VF_BAR	Section 10.3.5.1.3
0x0000000C + 0x10*n, n=0...16	VFMSIX_TVCTRL[n]	VF MSI-X Vector Control	VF_BAR	Section 10.3.5.1.4
0x00002000	VFMSIX_PBA	VF MSI-X PBA Structure	VF_BAR	Section 10.3.5.1.5

10.3.5 Detailed Register Description - VF BAR3

10.3.5.1 VF - MSI-X Table Registers

10.3.5.1.1 VF MSI-X Message Address Low - VFMSIX_TADD[n] (0x00000000 + 0x10*n, n=0...16; RW)

Message address for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTADD10	1:0	00b	RW	Message Address 1:0 For proper Dword alignment, software must always write zeros to these two bits; otherwise, the result is undefined. The state of these bits after reset must be 0b. These bits are permitted to be read-only or read/write.
MSIXTADD	31:2	0x0	RW	Message Address System-specified message lower address. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the lower portion of the Dword-aligned address (AD[31:02]) for the memory write transaction. This field is read/write.



10.3.5.1.2 VF MSI-X Message Address High - VFMSIX_TUADD[n] (0x00000004 + 0x10*n, n=0...16; RW)

Message upper address for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTUADD	31:0	0x0	RW	Message Upper Address System-specified message upper address bits. If this field is zero, Single Address Cycle (SAC) messages are used. If this field is non-zero, Dual Address Cycle (DAC) messages are used. This field is read/write.

10.3.5.1.3 VF MSI-X Message Data - VFMSIX_TMSG[n] (0x00000008 + 0x10*n, n=0...16; RW)

Message data for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MSIXTMSG	31:0	0x0	RW	Message Data System-specified message data. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the data driven on AD[31:0] during the memory write transaction's data phase. This field is read/write.

10.3.5.1.4 VF MSI-X Vector Control - VFMSIX_TVCTRL[n] (0x0000000C + 0x10*n, n=0...16; RW)

Vector control for MSI-X table entries.

Field	Bit(s)	Init.	Type	Description
MASK	0	1b	RW	Mask Bit 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 are still capable of sending an equivalent message unless they are also masked. This bit's state after reset is 1b (entry is masked).
RESERVED	31:1	0x0	RSV	Reserved. After reset, the state of these bits must be 0b. However, for potential future use, software must preserve the value of these reserved bits when modifying the value of other Vector Control bits. If software modifies the value of these reserved bits, the result is undefined.



10.3.5.1.5 VF MSI-X PBA Structure - VFMSIX_PBA (0x00002000; RO)

Pending bits for MSI-X PBA entries.

Field	Bit(s)	Init.	Type	Description
PENBIT	31:0	0x0	RO	MSI-X Pending Bits Each bit is set to 1b when the appropriate interrupt request is set and cleared to 0b when the appropriate interrupt request is cleared.



11.0 PCIe* Programming Interface

11.1 Overview

The X710-TM4 supports the following configuration register sets:

- PCI basic configuration registers (see [Section 11.2](#)).
- PCI and PCIe capabilities in the PCI configuration space (see [Section 11.3](#)).
 - Includes the PCIe capability structure (see [Section 11.3.5](#)).
- PCIe capabilities residing in the PCIe extended configuration space (see [Section 11.4](#)).
- SR-IOV VF configuration space (see [Section 11.5](#)).

11.1.1 Functions Mapping

The X710-TM4 is a multi-function device with the following characteristics:

- Up to 8 Physical Functions (PFs) in non-ARI mode and up to 16 functions in ARI mode.
- Up to 128 SR-IOV Virtual Functions (VFs)
 - Each PF can be allocated a different number of VFs in the range $\{0, \dots, 128\}$ as long as the total number of VFs does not exceed 128.

The following rules apply regarding allocation of PCI functions to LAN ports:

- Each PCI function is associated with a single LAN port as indicated in the PFGEN_PORTNUM.PORT_NUM register field.
- The number of PFs associated with an enabled LAN port might differ among ports and might be any number in the range $\{1, \dots, 8\}$

Note: NVM programming notes:

Set PFGEN_PORTNUM_CAR in NVM POR auto-load section.

Set PFGEN_PORTNUM in NVM CORER auto-load section.

Set PFGEN_PORTMDIO_NUM with the same values as PFGEN_PORTNUM in NVM CORER auto-load section.

The number of enabled physical functions might be deducted from the PFPCI_STATUS1.FUNC_VALID bits (one per PF). See [Section 4.5.4](#) on how enabled functions are determined.

The ARI capability enables interpretation of the device number part of the RID as part of the function number inside a device. Thus, a single device can span more than eight physical or virtual functions. [Table 11-1](#) and [Table 11-2](#) map the physical functions to PCI requester ID.



Table 11-1. RID Per PF - ARI Mode

PF#	B,D,F	Binary	Notes
PF 0	B,0,0	B,00000,000	PF #0
PF 1	B,0,1	B,00000,001	PF #1
PF 2	B,0,2	B,00000,010	PF #2
PF 15	B,1,7	B,00001,111	PF #15

The requester ID of a PF (bus, device, and function numbers) is captured in the PF_FUNC_RID register.

11.1.1.1 Support for Dynamic Changes

The X710-TM4 captures the bus number and device number per each configuration write request. However, a dynamic change of the bus number or device number is not supported. Rather, the PCIe link should be quiescent prior to such a change, including reception of all completion for previous requests.

11.1.2 Supported Features

Table 11-2 lists the PCI and PCIe capabilities supported per PCI function type. Some capabilities do not necessarily appear with each function or in all cases. See Section 11.3 and Section 11.4 for details on specific capabilities.

Table 11-2. PCI Capabilities Supported by Function

PCI Capability	PFs	VFs
PCI Configuration	Mandatory	Mandatory
Power Management	Yes	Yes
MSI	Yes	No
MSI-X	Yes	Yes
Vital Product Data (VPD)	Yes	No
PCIe	Yes	Yes
Advanced Error Reporting (AER)	Yes	Yes
Device Serial Number	Yes	No (N/A)
Alternative RID Interpretation (ARI)	Yes	Yes
Single Root I/O Virtualization (SR-IOV)	Yes	No (N/A)
TPH Requester	Yes	Yes
Access Control Services (ACS)	Yes	Yes
Secondary PCIe	Yes	No (N/A)



11.2 PCI Configuration Space

Configuration registers are assigned one of the attributes listed in the table that follows.

11.2.1 Register Attributes

The following table lists the register attributes used in this section.

RD/WR	Description
RO	Read-only register: Register bits are read-only and cannot be altered by software.
RW	Read-write register: Register bits are read-write and can be either set or reset.
RW1C	Read-only status, Write-1b-to-clear status register, Writing a 0b to RW1C bits has no effect.
ROS	Read-only register with sticky bits: Register bits are read-only and cannot be altered by software. Bits are neither initialized nor modified by PCIe in-band reset or FLR. Specific bits listed below are also not reset on PERST# when aux power consumption is enabled.
RWS	Read-write register: Register bits are read-write and can be either set or cleared by software to the desired state. Bits are neither initialized nor modified by PCIe in-band reset or FLR. Specific bits listed below are also not reset on PE_RST_N when aux power consumption is enabled.
RW1CS	Read-only status, Write-1b-to-clear status register: Register bits indicate status when read, a set bit, indicating a status event, can be cleared by writing a 1b to it. Writing a 0b to RW1C bits has no effect. Bits are neither initialized nor modified by PCIe in-band reset or FLR. Specific bits listed in the sections that follow are also not reset on PE_RST_N when aux power consumption is enabled.
HwInit	Hardware initialized: Register bits are initialized by firmware or hardware mechanisms such as pin strapping or serial NVM. Bits are read-only after initialization and can only be reset (for write-once by firmware) with the PWRGOOD signal.
RsvdP	Reserved and preserved: Reserved for future read/write implementations; software must preserve value read for writes to these bits.
RsvdZ	Reserved and zero: Reserved for future RW1C implementations; software must use 0b for writes to these bits.

11.2.2 Reset Rules

Reset of the PCI configuration space (including any capability lists) is per the PCIe specification. Several cases require special attention.

11.2.2.1 Sticky Registers

The following sticky register fields are also not reset on PE_RST_N when aux power consumption is enabled (AUX_PWR pin is set).

- PME related fields:
 - Power Management Capabilities register — PME_En bit
 - Power Management Capabilities register — PME_Status bit



- Device Control register — *Aux Power PM Enable* bit
- The function Requester ID
- AER related fields:
 - Uncorrectable Error Status register
 - Uncorrectable Error Mask register
 - Uncorrectable Error Severity register
 - Correctable Error Status register
 - Correctable Error Mask register
 - Advanced Error Capabilities and Control register
 - Header log

11.2.2.2 Reset on FLR

The following registers are not affected by FLR:

- The *Max Payload Size* field in the Device Control register
- The *Active State Link PM Control* field in the Link Control register
- The *Common Clock Configuration* field in the Link Control register
- The *Extended Sync* field in the Link Control register
- The *Link Equalization Request* field in the Link Status 2 register

11.2.3 PCI Configuration Space Summary

Table 11-3 lists the PCI configuration registers while their detailed description is given in the sections that follow. Fields that have meaningful default values are indicated in parenthesis — (**value**).

The PCI configuration space from address 0x40 on is allocated for PCI capability structures as described in Section 11.3. However, the region of 0x98-0x9F (8 bytes) is dedicated to an address/data port, which provides access to the device I/O address space (see Section 10.1.1.6 for more details).



Table 11-3. PCI Configuration Space - PF

Section	Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0	
Mandatory PCI Register	0x0	Device ID		Vendor ID		
	0x4	Status Register		Command Register		
	0x8	Class Code (0x020000/0x010000)			Revision ID	
	0xC	Reserved	Header Type (0x0/0x80)	Latency Timer	Cache Line Size (0x10)	
	0x10	Base Address Register 0				
	0x14	Base Address Register 1				
	0x18	Base Address Register 2				
	0x1C	Base Address Register 3				
	0x20	Base Address Register 4				
	0x24	Base Address Register 5				
	0x28	CardBus CIS Pointer (0x0000)				
	0x2C	Subsystem ID		Subsystem Vendor ID		
	0x30	Expansion ROM Base Address				
	0x34	Reserved			Cap Ptr (0x40)	
	0x38	Reserved				
	0x3C	Max Latency (0x00)	Min Grant (0x00)	Interrupt Pin (0x01...0x04)	Interrupt Line (0x00)	

11.2.4 Sharing Among PCI Functions

The X710-TM4 supports multiple PCI functions. As each function exposes a PCIe configuration space, each register and each field is either shared among the functions or is replicated per each PCI function. This section summarizes configuration sharing of the fixed PCI configuration space. See the description of each PCI capability structure for configuration sharing within it. Also, the description of each field describes special considerations regarding configuration sharing.

Table 11-4. Configuration Sharing of PCI Configuration Space

Field	Sub-field	Shared?	Replicated?	Comments
Vendor ID	Vendor ID	x		
Device ID	Device ID		x	
Command Register	I/O Access Enable		x	Issue UR per PF if disabled.
	Memory Access Enable		x	Issue UR per PF if disabled.
	Bus Master Enable		x	
	Parity Error Response		x	Enables certain error reporting per PF.
	SERR# Enable		x	Controls error reporting per PF.
Status Register	Interrupt Disable		x	Selection of interrupt method per PF.
	Interrupt Status		x	
	Capabilities List	x		Hardwired to 1b.
	Data Parity Reported / Master Data Parity Error		x	Reports poisoned packets per PF.



Table 11-4. Configuration Sharing of PCI Configuration Space

Field	Sub-field	Shared?	Replicated?	Comments
	Signaled Target Abort		x	Reports Completer Abort per PF.
	Received Target Abort		x	Reports receiving a Completer Abort per PF.
	Received Master Abort		x	Reports receiving an UR per PF.
	Signaled System Error		x	Reports Fatal / non-fatal message per PF.
	Detected Parity Error		x	Reports receiving a poisoned TLP per PF.
Revision Register		x		
Class Code Register			x	Per function type.
Cache Line Size Register			x	Does not affect device behavior.
Latency Timer		x		Hardwired to 0x00 in PCIe.
Header Type Register		x		
BIST		x		Not supported (0x00).
BARs	Memory BAR		x	
	I/O BAR		x	
	MSI-X BAR		x	See MSI-X capability.
I/O BAR mapping	IOADDR, IODATA		x	
Subsystem Vendor ID		x		
Subsystem ID			x	
Expansion ROM		x		Each PF has its own BAR.
Cap_Ptr Register		x		
Interrupt Line Register			x	Just store the register value.
Interrupt Pin Register			x	Separate interrupt number (A-D) per PF.
Min Grant		x		
Max Latency		x		

11.2.5 Mandatory PCI Configuration Registers – Except BARs

11.2.5.1 Vendor ID Register (0x0; RO)

This is a read-only register that has the same value for all PCI functions.

- Vendor ID is loaded from the NVM if the *GLPCI_CAPSUP.LOAD_DEV_ID* bit is set.
- The value for all PFs is loaded from the NVM *GLPCI_VENDORID.VENDOR_ID* field.



11.2.5.2 Device ID Register (0x2; RO)

This is a read-only register that identifies individual X710-TM4 PCI functions. All functions have the same default value of 0x154B, and can be auto-loaded from the NVM during initialization with different values for each function as well as the dummy function.

Device ID is loaded from the NVM according to the following rules:

- The Device ID is loaded from the NVM if the *GLPCI_CAPSUP.LOAD_DEV_ID* bit is set.
- The value of each PF is loaded to the respective *PFPCI_DEVID.PF_DEV_ID* field.

11.2.5.3 Command Register (0x4; RW)

Shaded bits are not used by this implementation and are hardwired to 0b. Each function has its own Command register (see [Table 11-4](#)).

Bit(s)	Initial Value	RW	Description
0	0b	RW (see comment)	I/O Access Enable. This bit is RO if an I/O BAR is not supported by the device.
1	0b	RW	Memory Access Enable.
2	0b	RW	Enable Mastering, also named Bus Master Enable (BME). <ul style="list-style-type: none"> • LAN functions RW field. • Dummy function RO as zero field.
3	0b	RO	Special Cycle Monitoring – Hardwired to 0b.
4	0b	RO	MWI Enable – Hardwired to 0b.
5	0b	RO	Palette Snoop Enable – Hardwired to 0b.
6	0b	RW	Parity Error Response.
7	0b	RO	Wait Cycle Enable – Hardwired to 0b.
8	0b	RW	SERR# Enable.
9	0b	RO	Fast Back-to-Back Enable – Hardwired to 0b.
10	0b	RW	Interrupt Disable. When set, devices are prevented from generating legacy interrupt messages. RO as 0b for a dummy function.
15:11	0b	RO	Reserved.

11.2.5.4 Status Register (0x6; RO)

Shaded bits are not used by this implementation and are hardwired to 0b. Each function has its own Status register.

Bits	Initial Value	RW	Description
2:0	0b		Reserved.
3	0b	RO	Interrupt Status. ¹



Bits	Initial Value	RW	Description
4	1b	RO	New Capabilities: Indicates that a device implements extended capabilities. The X710-TM4 sets this bit and implements a capabilities list to indicate that it supports PCI and PCIe capabilities.
5	0b		66 MHz Capable – Hard wired to 0b.
6	0b		Reserved.
7	0b		Fast Back-to-Back Capable – Hard wired to 0b.
8	0b	RW1C	Data Parity Reported.
10:9	00b		DEVSEL Timing – Hard wired to 0b.
11	0b	RW1C	Signaled Target Abort.
12	0b	RW1C	Received Target Abort.
13	0b	RW1C	Received Master Abort.
14	0b	RW1C	Signaled System Error.
15	0b	RW1C	Detected Parity Error.

1. The *Interrupt Status* field is a RO field that indicates that an interrupt message is pending internally to the device.

11.2.5.5 Revision Register (0x8; RO)

The default revision ID of this device is 0x01. The default value is readable through the *GLPCI_DREVID* register. The value in the Revision register is a logic XOR between the default value and a value loaded from the NVM, reflected via the *GLPCI_REVID* register.

11.2.5.6 Class Code Register (0x9; RO)

The class code is a read-only value that identifies the device functionality:

- Class Code = 0x020000 (Ethernet adapter) if NVM->*Storage Class* = 0b.
- Class Code = 0x010000 (SCSI storage device) if NVM->*Storage Class* = 1b.

In the dummy function the class code equals to 0xFF0000.

Device default value is class code of an Ethernet adapter. The value is overwritten from the NVM. It is loaded to the RO PFPCI_CLASS register.

11.2.5.7 Cache Line Size Register (0xC; RW)

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. All functions are initialized to the same value of 0x00 (specification required).

11.2.5.8 Latency Timer (0xD; RO)

Not used. Hardwired to 0b.



11.2.5.9 Header Type Register (0xE; RO)

This indicates if a device is single- or multi-function:

- 0x00 — A single function is enabled
- 0x80 — At least two functions are enabled

Note:

- Dummy functions are considered as regular functions in this regard.
- SR-IOV VFs are not counted in the setting of the Header type register
- Functions might be disabled during the power on reset flow (through strapping pins, SMASH/CLP commands, NC-SI commands) affecting this bit. See [Section 4.5.4](#).

11.2.5.10 Subsystem vendor ID register (0x2C; RO)

This value is loaded from the NVM if the *GLPCI_CAPSUP.LOAD_SUBSYS_ID* bit is set. A value of 0x8086 is the default for this field. It can be read through the *GLPCI_SUBVENID* register.

11.2.5.11 Subsystem ID Register (0x2E; RO)

This value is loaded from the NVM if the *GLPCI_CAPSUP.LOAD_SUBSYS_ID* bit is set. Each PF is loaded to the respective *PFPCI_SUBSYSID.PF_SUBSYS_ID* field.

11.2.5.12 Capabilities Pointer Register (0x34; RO)

The *Capabilities Pointer* field (*Cap_Ptr*) is an 8-bit field that provides an offset in the X710-TM4's PCI configuration space for the location of the first item in the capabilities linked list. The X710-TM4 supports this field and implements a capabilities list. Its value is 0x40, which is the address of the first entry: PCI power management.

11.2.5.13 Interrupt Line Register (0x3C; RW)

Read/write register programmed by software to indicate which of the system interrupt request lines the X710-TM4's interrupt pin is bound to. Refer to the PCI definition for more details.

11.2.5.14 Interrupt Pin Register (0x3D; RO)

Read-only register. — A value of 0x1...0x4 indicates that this function implements a legacy interrupt on INTA#...INTD# respectively. Loaded from the NVM. It can be read through the RO *PFPCI_CNF* register. Device default value is 0x0.

If only a single function is enabled, the *Interrupt Pin* field of the enabled function reports INTA# usage. Reports a value of 0x0 (function uses no legacy interrupt message) for a dummy function.



11.2.5.15 MIN_GNT and MAX_LAT (0x3E; RO)

Not used. Hardwired to 0b.

11.2.6 Mandatory PCI Configuration Registers – BARs

11.2.6.1 Memory and I/O BARs (0x10...0x27; RW)

BARs are used to map the X710-TM4 register space of the device functions. The X710-TM4 has a memory BAR, an I/O BAR (optional) an MSI-X BAR as listed in Table 11-5. The BARs location and sizes are listed in Table 11-6. The fields within each BAR are then listed in Table 11-7.

Table 11-5. X710-TM4 BAR description

Mapping Windows	Mapping Description
Memory BAR	The internal registers memories and external Flash devices are accessed as direct memory mapped offsets from the BAR. Software can access a Dword or 64 bits.
I/O BAR	All internal registers and memories can be accessed using I/O operations. There are two 4-byte registers in the I/O mapping window: Addr Reg and Data Reg accessible as Dword entities. The I/O BAR is supported depending on the NVM configuration. Device default value is that an I/O BAR is not supported. The state of the I/O BAR is exposed in the PFPCI_CNF register. This BAR is not present in dummy functions.
MSI-X BAR	The MSI-X vectors and PBA structures are accessed as direct memory mapped offsets from the MSI-X BAR. Software can access Dword entities. This BAR is not present in dummy functions.

Table 11-6. X710-TM4 Base Address Setting in 64-bit BARs Mode

BAR	Addr	31	5	4	3	2	1	0
0	0x10	Memory CSR + Flash BAR Low: 31:24 = RW; 23:18 = RW or 0b; 17:4 = 0b			0/1	1	0	0
1	0x14	Memory CSR + Flash BAR High (RW)						
2	0x18	IO BAR (RW – 31:5) (optional)		0	0	0	0	1
3	0x1C	MSI-X BAR Low (RW – 31:15; RO 0b – 14:4)			0/1	1	0	0
4	0x20	MSI-X BAR High (RW)						
5	0x24	Reserved (RO – 0)						

**Table 11-7. Base Address Registers' Fields**

Field	bits	RW	Description	
Memory and I/O Space Indication	0	RO	0b = Indicates memory space. 1b = Indicates I/O.	
Memory Type	2:1	RO	This field is loaded from the NVM. Hardware default is 64-bit. The field is exposed in the GLPCI_LBARCTRL register. 00b = 32-bit BAR. 10b = 64-bit BAR.	
Prefetch Memory	3	RO	This bit is loaded from NVM. This bit should be set only on systems that do not generate prefetchable cycles. Device default is 1b (prefetchable). It is exposed in the GLPCI_LBARCTRL register. 0b = Non-prefetchable space. 1b = Prefetchable space.	
Address Space (low register for 64-bit memory BARs)	31:4	RW	The length of the RW bits and RO 0b bits depend on the mapping window sizes. Initial value of the RW fields is 0x0.	
			Mapping Window	RO bits
			Memory space for CSRs and Flash memory access.	16:4 for 128 KB 17:4 for 256 KB and so on...
			MSI-X space is 32 KB.	14:4
			I/O space size is 32 bytes.	4:0

11.2.6.2 Expansion ROM Base Address Register (0x30; RW)

This register is used to define the address and size information for boot-time access to the optional Flash memory. This register returns a zero value for functions without an expansion ROM window and for dummy functions.

The expansion ROM BAR is disabled through the PFPCI_CNF.EXROM_DIS register field.

Field	Bit(s)	RW	Initial Value	Description
En	0	RW	0b	1b = Enables expansion ROM access. 0b = Disables expansion ROM access.
Reserved	10:1	R	0b	Always read as 0b. Writes are ignored.
Address	31:11	RW	0b	The number of bits that are not hardwired to 0b is determined by the value of the GLPCI_LBARCTRL.EXROM_SIZE register field, loaded from the NVM.

11.3 Capabilities in PCI Configuration Space

The first entry of the PCI capabilities link list is pointed to by the Cap_Ptr register. [Table 11-8](#) lists the capabilities supported by the X710-TM4 that reside in the PCI configuration space.



Table 11-8. PCI Capabilities List

Address range	Item	Cases Where Capability Does Not Exist	Next Pointer
0x40:4F	Power Management	None (always exists).	0x50 / 0xA0
0x50:6F	MSI	Dummy function.	0x70 / 0xA0
0x70:8F	MSI-X	1. PFPCI_CNFG.MSI_EN is 0b. 2. Dummy function.	0xA0
0xA0:DF	PCIe	None (always exists).	0xE0 / 0x00
0xE0:0xEF	VPD	1. VPD Enable bit (GLPCI_CAPCTRL.VPD_EN) is cleared. 2. Dummy function.	0x00

11.3.1 PCI Power Management Capability

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x40	Power Management Capabilities		Next Pointer	Capability ID (0x01)
0x44	Data	Bridge Support Extensions	Power Management Control & Status	

Table 11-9 lists the sharing of the Power Management Capability registers among the different PCI functions.

Table 11-9. Sharing the Power Management Capability Registers

Field	Sub-field	Shared?	Replicated?	Comments
Capability ID		x		
Next Pointer			x	
Power Management Capabilities	PME_Support	x		
	D2_Support	x		
	D1_Support	x		
	AUX Current	x		
	DSI	x		
	PME Clock	x		Hardwired to 0b
	Version	x		
Power Management Control / Status	PME_Status		x	
	Data_Scale	x		
	Data_Select		x	
	PME_En		x	



Table 11-9. Sharing the Power Management Capability Registers

Field	Sub-field	Shared?	Replicated?	Comments
	No_Soft_Reset	x		
	PowerState		x	
Data Register			x	

11.3.1.1 Capability ID Register (0x40; RO)

This field equals 0x01 indicating the linked list item as being the PCI Power Management registers.

11.3.1.2 Next Pointer Register (0x41; RO)

This field provides an offset to the next capability item in the capability list. See Table 11-8 for possible values of the next pointer register.

11.3.1.3 Power Management Capabilities – PMCR (0x42; RO)

This field describes the device functionality during the power management states as listed in the following table.

Bits	Default	RW	Description
15:11	01001b	RO	PME_Support. This 5-bit field indicates the power states in which the function can generate a PME event. Condition functionality values are as follows: <ul style="list-style-type: none"> No AUX Pwr PME at D0 and D3hot = 01001b AUX Pwr PME at D0, D3hot, and D3cold = 11001b Note: For dummy function, this field is RO - zero.
10	0b	RO	D2_Support – The X710-TM4 does not support the D2 state.
9	0b	RO	D1_Support – The X710-TM4 does not support the D1 state.
8:6	000b	RO	AUX Current – Required current defined in the Data register.
5	1b	RO	DSI – The X710-TM4 requires its device driver to be executed following a transition to the D0 uninitialized state.
4	0b	RsvdP	Reserved.
3	0b	RO	PME_Clock – Disabled. Hardwire to 0b.
2:0	011b	RO	Version – The X710-TM4 complies with the PCI PM specification revision 1.2.

11.3.1.4 Power Management Control / Status Register – PMCSR (0x44; RW)

This register (listed in the following table) is used to control and monitor power management events in the device.



Bits	Default	RW	Description
15	0b (at power up)	RW1CS	PME_Status. This bit is set to 1b when the function detects a wake-up event independent of the state of the <i>PME_En</i> bit. Writing a 1b clears this bit.
14:13	00b	RO	Data_Scale. This field indicates the scaling factor that's used when interpreting the value of the Data register. This field is loaded from the NVM (through the GLPCI_PWRDATA.DATA_SCALE register) for legal values of Data_Select [0, 3, 4, 7, (and 8 for function 0)]. The normal value is 01b (indicating 0.1 watt/units). Reserved (00b) for any other values of Data_Select.
12:9	0000b	RW	Data_Select. This 4-bit field is used to select which data is to be reported through the Data register and <i>Data_Scale</i> field.
8	0b (at power up)	RWS	PME_En. Writing a 1b to this register enables wake up.
7:4	0000b	RO	Reserved.
3	1b	RO	No_Soft_Reset. This bit is always set to 1b to indicate that the X710-TM4 does not perform an internal reset upon a transition from D3hot to D0 via software control of the <i>PowerState</i> bits. Configuration context is maintained when performing the soft reset. Upon transition from the D3hot to the D0 state, an initialization sequence is not needed in order to return the X710-TM4 to the D0 Initialized state.
2	0b	RO	Reserved for PCIe.
1:0	00b	RW	PowerState. This field is used to set and report the power state of a function as follows: 00b = D0. 01b = D1 (cycle ignored if written with this value). 10b = D2 (cycle ignored if written with this value). 11b = D3

11.3.1.5 PMCSR_BSE Bridge Support Extensions Register (0x46; RO)

This register is not implemented in the X710-TM4; values set to 0x00.

11.3.1.6 Data Register (0x47; RO)

This optional register is used to report power consumption and heat dissipation. The reported register is controlled by the *Data_Select* field in the PMCSR; the power scale is reported in the *Data_Scale* field in the PMCSR. The data for this field is loaded from the NVM with a default value of 0x00. It is exposed through the GLPCI_PWRDATA register.

The values for the X710-TM4 functions are as follows (the relevant column is selected based on the value of the *Data_Select* field):



Field	D0 (Consume/ Dissipate)	D3 (Consume/ Dissipate)	Common
Data_Select	(0x0/0x4)	(0x3/0x7)	(0x8)
Function 0	Loaded from NVM	Loaded from NVM	Multi-function value: Loaded from the NVM Single-function value: 0x00
Other functions	Loaded from NVM	Loaded from NVM	0x00

Note: For other Data_Select values the Data register output is reserved (0x00).

11.3.2 MSI Capability

This structure is enabled when the PFPCI_CNF.MSI_EN bit is set for the function.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x50	Message Control (0x0080)		Next Pointer	Capability ID (0x05)
0x54	Message Address			
0x58	Message Upper Address			
0x5C	Reserved		Message Data	
0x60	Mask Bits			
0x64	Pending Bits			

Table 11-10 lists configuration sharing of the MSI Capability registers among the different PCI functions.

Table 11-10. Configuration Sharing of the MSI Capability

Field	Sub-field	Shared?	Replicated?	Comments
Capability ID		x		
Next Pointer			x	
Message Control	MSI Enable		x	
	Multiple Messages Capable	x		
	Multiple Message Enable		x	
	64-bit Capable	x		
	MSI Per-vector Masking	x		
Message Address Low			x	
Message Address High			x	



Table 11-10. Configuration Sharing of the MSI Capability

Field	Sub-field	Shared?	Replicated?	Comments
Message Data			x	
Mask Bits			x	
Pending Bits			x	

11.3.2.1 Capability ID Register (0x50; RO)

This field equals 0x05 indicating that the linked list item as being the MSI registers.

11.3.2.2 Next Pointer Register (0x51; RO)

This field provides an offset to the next capability item in the capability list. See [Table 11-8](#) for possible values of the next pointer register.

11.3.2.3 Message Control Register (0x52; RW)

Bits	Default	RW	Description
0	0b	RW	MSI Enable. 1b = Message Signaled Interrupts. The X710-TM4 generates an MSI for interrupt assertion instead of INTx signaling.
3:1	000b	RO	Multiple Messages Capable. The X710-TM4 indicates a single requested message per function.
6:4	000b	RW	Multiple Message Enable. Software writes to this field to indicate the number of allocated vectors Since The X710-TM4 requests a single vector in the <i>Multiple Messages Capable</i> field, software is expected to write 000b to this field.
7	1b	RO	64-bit Capable. A value of 1b indicates that the X710-TM4 is capable of generating 64-bit message addresses.
8	1b ¹	RO	MSI Per-vector Masking. A value of 0b indicates that the X710-TM4 is not capable of per-vector masking. A value of 1b indicates that the X710-TM4 is capable of per-vector masking. Exposed through the GLPCI_CAPSUP register.
15:9	0b	RO	Reserved. Reads as 0b

1. Value loaded from the NVM.

11.3.2.4 Message Address Low Register (0x54; RW)

Written by the system to indicate the lower 32 bits of the address to use for the MSI memory write transaction. The lower two bits always return 0b regardless of the write operation.



11.3.2.5 Message Address High Register (0x58; RW)

Written by the system to indicate the upper 32 bits of the address to use for the MSI memory write transaction.

11.3.2.6 Message Data Register (0x5C; RW)

Written by the system to indicate the lower 16 bits of the data written in the MSI memory write Dword transaction. The upper 16 bits of the transaction are written as 0b.

11.3.2.7 Mask Bits Register (0x60; RW)

The Mask Bits and Pending Bits registers enable software to disable or defer message sending on a per-vector basis. Because the X710-TM4 only supports one message, only bit 0 of these registers are implemented.

Bits	Default	RW	Description
0	0b	RW	MSI Vector 0 Mask. If set, the X710-TM4 is prohibited from sending MSI messages.
31:1	0x0	RO	Reserved.

11.3.2.8 Pending Bits Register (0x64; RW)

Bits	Default	RW	Description
0	0b	RO	If set, the X710-TM4 has a pending MSI message.
31:1	0x0	RO	Reserved.

11.3.3 MSI-X Capability

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.

A BAR is allocated for the MSI-X structures, described in [Section 11.2.6](#). 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 can be either 32-bit or is 64-bit.

The number of MSI-X vectors per PF (denoted as N) varies with the number of physical and virtual functions.

The MSI-X BAR is 32 KB long.



The location and size of the MSI-X vector table and the MSI-X pending bits table are determined as follows:

- MSI-X vector table:
 - The MSI-X table structure (Section 11.3.3.2) 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.
 - Starts at offset 0x0000 from start of BAR.
 - Contains the MSI-X vectors for the PF. The maximum value of N is 129 per PF (for the case of up to 4 PFs).
 - The vectors start with the Vector 0 (one per PF), followed by the other vectors allocated to the PF.
- MSI-X Pending Bits table:
 - The PBA structure [Section 11.3.3.2.2] 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.
 - Starts at offset 0x1000 (4KB) from start of BAR.
 - Contains the pending bits for the PF. The PF may be allocated a multiple of 64-bit registers. The total number of 32-bit registers is per the number of MSI-X vectors. The maximum number of registers is 6 (for the case of 129 vectors).
 - The bits start with the Vector 0 bit (one per PF), followed by bits for the other vectors allocated to the PF.

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.

11.3.3.1 Capability Structure

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x70	Message Control (0x00090)		Next Pointer	Capability ID (0x11)
0x74	Table Offset			
0x78	PBA Offset			

Table 11-11 lists configuration sharing of the MSI-X capability registers among the different PCI functions.



Table 11-11. Configuration Sharing of the MSI-X Capability

Field	Sub-field	Shared?	Replicated?	Comments
Capability ID		x		
Next Pointer			x	
Message Control	Table Size	x		
Function Mask			x	
MSI-X Enable			x	
MSI-X Table Offset	Table BIR		x	
	Table Offset		x	
MSI-X Pending Bit Array	PBA BIR		x	
	PBA Offset		x	
MSI-X Table			x	
MSI-X PBA Structure			x	

11.3.3.1.1 Capability ID Register (0x70; RO)

This field equals 0x11 indicating that the linked list item as being the MSI-X registers.

11.3.3.1.2 Next Pointer Register (0x71; RO)

This field provides an offset to the next capability item in the capability list. See [Table 11-8](#) for possible values of the next pointer register.



11.3.3.1.3 Message Control Register (0x72; RW)

Bits	Default	RW	Description
10:0	0x80 (129 vectors)	RO	Table Size. System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. N varies with the number of physical functions. This field is loaded from the NVM. It is reflected in the GLPCI_CNF2.MSI_X_PF_N CSR field.
13:11	0x0	RO	Always returns 0b on a read. A write operation has no effect.
14	0b	RW	Function Mask. If 1b, all of the vectors associated with the function are masked, regardless of their per-vector <i>Mask</i> bit states. If 0b, each vector's <i>Mask</i> bit determines whether the vector is masked or not. Setting or clearing the MSI-X <i>Function Mask</i> bit has no effect on the state of the per-vector <i>Mask</i> bits.
15	0b	RW	MSI-X Enable. If 1b and the MSI <i>Enable</i> 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. If 0b, the function is prohibited from using MSI-X to request service.

11.3.3.1.4 MSI-X Table Offset Register (0x74; RW)

Bits	Default	RW	Description
2:0	0x3	RO	Table BIR. Indicates which one of a function's BARs, beginning at 0x10 in the configuration space, is used to map the function's MSI-X table into the memory space. While BIR values: 0...5 correspond to BARs 0x10:0x 24, respectively.
31:3	0x000	RO	Table Offset. Used as an offset from the address contained in one of the function's BARs to point to the base of the MSI-X table. The lower three <i>Table BIR</i> bits are masked off (set to 0b) by software to form a 32-bit Qword-aligned offset. Note that this field is read only.

11.3.3.1.5 MSI-X Pending Bit Array – PBA Offset (0x78; RW)

Bits	Default	RW	Description
2:0	0x3	RO	PBA BIR. Indicates which one of a function's BARs, beginning at 0x10 in the configuration space, is used to map the function's MSI-X PBA into the memory space. While BIR values: 0...5 correspond to BARs 0x10:0x 24, respectively.
31:3	0x200	RO	PBA Offset. 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. This field is read only.



11.3.3.2 PF MSI-X Table Structure

The MSI-X table is made of two structures:

- MSI-X Vector Table
- MSI-X Pending Bits Table

Both are described in the sections that follow and in more detail in [Section 10.0](#).

11.3.3.2.1 MSI-X Vector Table

Dword3 – MSIXCTRL	Dword2 – MSIXTMSG	Dword1 – MSIXTUADD	Dword0 – MSIXTADD	Entry Number	BAR 3 – Offset
Vector Control	Msg Data	Msg Upper Addr	Msg Lower Addr	0	Base (0x0000)
Vector Control	Msg Data	Msg Upper Addr	Msg Lower Addr	1	Base + 1*16
Vector Control	Msg Data	Msg Upper Addr	Msg Lower Addr	2	Base + 2*16
...	
Vector Control	Msg Data	Msg Upper Addr	Msg Lower Addr	(N-1)	Base + (N-1)*16

11.3.3.2.2 MSI-X Pending Bits Table

Field	Bit(s)	Init Val.	Description
PENBIT	31:0	0x0	MSI-X Pending Bits. Each bit is set to 1b when the appropriate interrupt request is set and cleared to 0b when the appropriate interrupt request is cleared.

11.3.4 VPD Capability

The X710-TM4 supports access to a VPD structure stored in the NVM using the following set of registers. A single VPD structure is provided, accessible through any of the physical functions.

Initial values of the configuration registers are marked in parenthesis.

Note: The VPD structure is available through all physical functions. As the interface is common to the functions, accessing the VPD structure of one function while an access to the NVM is in process on another function can yield to unexpected results.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0xE0	VPD Address		Next Pointer	Capability ID (0x03)
0xE4	VPD Data			



11.3.4.1 Capability ID Register (0xE0; RO)

This field equals 0x3 indicating the linked list item as being the VPD registers.

11.3.4.2 Next Pointer Register (0xE1; RO)

Offset to the next capability item in the capability list. See Table 11-8 for possible values of the next pointer register.

11.3.4.3 VPD Address Register (0xE2; RW)

Word-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.

Bits	Default	Rd/Wr	Description
14:0	X	RW	Address. 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.
15	0b	RW	F. A 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 is written to the NVM. The VPD address and data should not be modified before the action is done.

11.3.4.4 VPD Data Register (0xE4; RW)

VPD read/write data.

Bits	Default	Rd/Wr	Description
31:0	X	RW	VPD Data. VPD data can be read or written through this register. The LSB 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.

11.3.5 PCIe Capability Structure

The X710-TM4 implements the PCIe capability structure linked to the legacy PCI capability list for endpoint devices as follows:



Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0xA0	PCIe Capability Register (0x0002)		Next Pointer	Capability ID (0x10)
0xA4	Device Capability			
0xA8	Device Status		Device Control	
0xAC	Link Capability			
0xB0	Link Status		Link Control	
0xB4	Reserved			
0xB8	Reserved		Reserved	
0xBC	Reserved			
0xC0	Reserved		Reserved	
0xC4	Device Capability 2			
0xC8	Reserved		Device Control 2	
0xCC	Link Capabilities 2 Register			
0xD0	Link Status 2		Link Control 2	
0xD4	Reserved			
0xD8	Reserved		Reserved	

Table 11-12 lists configuration sharing of the PCIe Capability registers among the different PCI functions.

Table 11-12. Configuration Sharing of the PCIe Capability

Field	Sub-field	Shared?	Replicated?	Comments
Capability ID		x		
Next Pointer			x	
PCIe Capabilities		x		
Device Capabilities	Max Payload Size Supported	x		
	Phantom Functions Supported	x		Not supported.
	Extended Tag Field Supported	x		
	Endpoint L0s Acceptable Latency	x		
	Endpoint L1 Acceptable Latency	x		
	Function Level Reset Capability	x		
Device Control	Correctable Error Reporting Enable		x	
	Non-Fatal Error Reporting Enable		x	
	Fatal Error Reporting Enable		x	



Table 11-12. Configuration Sharing of the PCIe Capability

Field	Sub-field	Shared?	Replicated?	Comments
	Unsupported Request Reporting Enable		x	
	Enable Relaxed Ordering		x	
	Max Payload Size		x	Use a minimum of all configured values. In ARI mode, use value in function 0.
	Extended Tag Field Enable		x	
	Aux Power PM Enable		x	Same policy for all PFs (logical OR of the PFs' bits).
	Enable No Snoop		x	
	Max Read Request Size		x	Use a minimum of all configured values.
	Initiate Function Level Reset		x	
Device Status	Correctable Detected		x	
	Non-Fatal Error Detected		x	
	Fatal Error Detected		x	
	Unsupported Request Detected		x	
	Aux Power Detected	x		
	Transactions Pending		x	
Link Capabilities	Supported Link Speeds	x		
	Max Link Width	x		
	Active State Link PM Support	x		
	L0s Exit Latency	x		
	L1 Exit Latency	x		
	Clock Power Management	x		
	Port Number	x		
Link Control	Active State Link PM Control		x	Same policy for all PFs (logical AND of the PFs' bits). In ARI mode, use value in function 0.
	Read Completion Boundary (RCB)		x	
	Common Clock Configuration		x	Same policy for all PFs (logical AND of the PFs' bits). In ARI mode, use value in function 0.
	Extended Sync		x	Same policy for all PFs (logical OR of the PFs' bits).
Link Status	Current Link Speed	x		
	Negotiated Link Width	x		
	Slot Clock Configuration	x		
Device Capabilities 2	Completion Timeout Ranges Supported	x		
	Completion Timeout Disable Supported	x		
	Reserved	x		



Table 11-12. Configuration Sharing of the PCIe Capability

Field	Sub-field	Shared?	Replicated?	Comments
	TPH Completer Supported	x		
	Extended Fmt Field Supported	x		
	OBFF Supported	x		
Device Control 2	Completion Timeout Value		x	Completion timeout decision per PF or use the largest configured value among PFs.
	Completion Timeout Disable		x	Completion timeout mechanism enabled per PF.
	IDO Request Enable		x	
	IDO Completion Enable		x	
	Reserved		x	PF0 only. RsvdP on other functions.
	OBFF Enable		x	PF0 only. RsvdP on other functions.
Link Capabilities 2		x		
Link Control 2		x		PF0 only. RsvdP on other functions.
Link Status 2		x		

11.3.5.1 Capability ID Register (0xA0; RO)

This field equals 0x10 indicating that the linked list item as being the PCIe Capabilities registers.

11.3.5.2 Next Pointer Register (0xA1; RO)

Offset to the next capability item in the capability list. See Table 11-8 for possible values of the next pointer register.

11.3.5.3 PCIe Capabilities Register (0xA2; RO)

The PCIe Capabilities register identifies PCIe device type and associated capabilities.

Bits	Default	RW	Description
3:0	0010b	RO	Capability Version. Indicates the PCIe capability structure version. The X710-TM4 supports PCIe version 2 (loaded from NVM). It is reflected in the GLPCI_CAPSUP register.
7:4	0000b	RO	Device/Port Type. Indicates the type of PCIe functions. All functions are native PCI functions with a value of 0000b.
8	0b	RO	Slot Implemented. The X710-TM4 does not implement slot options. Therefore, this field is hardwired to 0b.
13:9	00000b	RO	Interrupt Message Number. This field is hardwired to 0x0 and is assumed to be irrelevant for endpoints.
15:14	00b	RO	Reserved.



11.3.5.4 Device Capabilities Register (0xA4; RO)

This register identifies the PCIe device specific capabilities.

Bits	Rd/Wr	Default	Description
2:0	RO	010b	Max Payload Size Supported. This field indicates the maximum payload that the X710-TM4 can support for TLPs. It is loaded from the NVM with a default value of 2 KB. It is loaded via the <i>GLPCI_LINKCAP</i> register.
4:3	RO	00b	Phantom Function Supported. Not supported by the X710-TM4.
5	RO	1b	Extended Tag Field Supported. Maximum supported size of the <i>Tag</i> field. The X710-TM4 supports an 8-bit <i>Tag</i> field for all functions.
8:6	RO	011b	Endpoint L0s Acceptable Latency. This field indicates the acceptable latency that the X710-TM4 can withstand due to the transition from the L0s state to the L0 state. All functions share the same value loaded from the NVM. The default value of 011b denotes a maximum 512 ns.
11:9	RO	110b	Endpoint L1 Acceptable Latency. This field indicates the acceptable latency that the X710-TM4 can withstand due to the transition from L1 state to the L0 state. The default value of 110b denotes a maximum of 64 μ s. All functions share the same value loaded from the NVM. It is reflected in the <i>GLPCI_PMSUP</i> register.
12	RO	0b	Hardwired in the X710-TM4 to 0b for all functions.
13	RO	0b	Hardwired in the X710-TM4 to 0b for all functions.
14	RO	0b	Hardwired in the X710-TM4 to 0b for all functions.
15	RO	1b	Role Based Error Reporting. Hardwired in the X710-TM4 to 1b for all functions.
17:16	RO	000b	Reserved 0b.
25:18	RO	0x00	Slot Power Limit Value. Hardwired in the X710-TM4 to 0x00 for all functions.
27:26	RO	00b	Slot Power Limit Scale. Hardwired in the X710-TM4 to 0b for all functions.
28	RO	1b	Function Level Reset Capability – A value of 1b indicates the function supports the optional FLR mechanism.
31:29	RO	0000b	Reserved.

11.3.5.5 Device Control Register (0xA8; RW)

This register controls the PCIe specific parameters.

Bits	RW	Default	Description
0	RW	0b	Correctable Error Reporting Enable. Enable error report.
1	RW	0b	Non-Fatal Error Reporting Enable. Enable error report.
2	RW	0b	Fatal Error Reporting Enable. Enable error report.
3	RW	0b	Unsupported Request Reporting Enable. Enable error report.
4	RW	1b	Enable Relaxed Ordering. If this bit is set, the X710-TM4 is permitted to set the <i>Relaxed Ordering</i> bit in the <i>Attribute</i> field of write transactions that do not need strong ordering. Refer to Section 3.1.2.7.2 for more details.



Bits	RW	Default	Description
7:5	RW	000b (128 bytes)	Max Payload Size. This field sets the maximum TLP payload size for the X710-TM4 functions. As a receiver, the X710-TM4 must handle TLPs as large as the set value. As a transmitter, the X710-TM4 must not generate TLPs exceeding the set value. In ARI mode, <i>Max Payload Size</i> is determined solely by the field in function 0 (even when it is a dummy function) while it is meaningless in the other function(s).
8	RW	1b	Extended Tag Field Enable. The X710-TM4 uses 8-bit tags when this bit is set and a 5-bit tag when disabled.
9	RO	0b	Phantom Functions Enable. Not implemented in the X710-TM4.
10	RWS	0b	Aux Power PM Enable. When set, enables the X710-TM4 to draw AUX power independent of PME AUX power. The X710-TM4 is a multi-function device, therefore allowed to draw AUX power if at least one of the functions has this bit set.
11	RO	0b	Enable No Snoop. Hardwired to 0b as the X710-TM4 never sets the no snoop attribute in a TLP.
14:12	RW	010b	Max Read Request Size. This field sets maximum read request size for the X710-TM4 as a requester. 000b = 128 bytes. 001b = 256 bytes. 010b = 512 bytes. 011b = 1024 bytes. 100b = 2048 bytes. 101b = 4096 bytes 110b = Reserved. 111b = Reserved.
15	RW	0b	Initiate FLR. A write of 1b initiates FLR to the function. The value read by software from this bit is always 0b.

11.3.5.6 Device Status Register (0xAA; RW1C)

This register provides information about PCIe device specific parameters.

Bits	RW	Default	Description
0	RW1C	0b	Correctable Detected. Indicates status of correctable error detection.
1	RW1C	0b	Non-Fatal Error Detected. Indicates status of non-fatal error detection.
2	RW1C	0b	Fatal Error Detected. Indicates status of fatal error detection.
3	RW1C	0b	Unsupported Request Detected. Indicates that the X710-TM4 received an unsupported request. This field is separate per PF. However, in case where an error cannot be associated with a PF, this bit is set in all PFs and VFs.
4	RO	0b	Aux Power Detected. 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.
5	RO	0b	Transactions Pending. Indicates whether the X710-TM4 has ANY transactions pending. Transactions include completions for any outstanding non-posted request for all used traffic classes.
15:6	RsvdZ	0x00	Reserved.

11.3.5.7 Link Capabilities Register (0xAC; RO)

This register identifies PCIe link-specific capabilities.



Bits	RW	Default	Description
3:0	RO	0x4	<p>Max Link Speed. This field indicates the maximum Link speed of the associated port.</p> <p>The encoding is the binary value of the bit location in the supported link speeds vector (in the Link Capabilities 2 register) that corresponds to the maximum link speed.</p> <p>For example, a value of 0011b in this field indicates that the maximum link speed is that corresponding to bit 2 in the supported link speeds vector, which is 8.0 GT/s.</p> <p>Multi-function devices associated with an upstream port must report the same value in this field for all functions.</p>
9:4	RO	0x08	<p>Maximum Link Width. Indicates the maximum link width. The X710-TM4 supports a x1, x4, and x8-link width. This field is loaded from the NVM and is reflected in the GLPCI_LINKCAP register.</p> <p>Defined encoding:</p> <p>000000b = Reserved.</p> <p>000001b = x1.</p> <p>000010b = Reserved.</p> <p>000100b = x4.</p> <p>001000b = x8.</p>
11:10	RO	11b	<p>Active State Link PM Support. Indicates the level of the active state of power management supported in the X710-TM4. Defined encodings are:</p> <p>00b = No ASPM support.</p> <p>01b = L0s entry supported.</p> <p>10b = L1 supported.</p> <p>11b = L0s and L1 supported.</p> <p>All functions share the same value loaded from the NVM. It is reflected in the GLPCI_PMSUP register.</p>
14:12	RO	101b	<p>L0s Exit Latency. Indicates the exit latency from L0s to L0 state.</p> <p>000b = Less than 64 ns.</p> <p>001b = 64 ns – 128 ns.</p> <p>010b = 128ns – 256 ns.</p> <p>011b = 256 ns – 512 ns.</p> <p>100b = 512 ns – 1 μs.</p> <p>101b = 1 μs – 2 μs.</p> <p>110b = 2 μs – 4 μs.</p> <p>111b = Reserved.</p> <p>All functions share the same value loaded from the NVM. It is reflected in the GLPCI_PMSUP register.</p>
17:15	RO	010b	<p>L1 Exit Latency. Indicates the exit latency from L1 to L0 state.</p> <p>000b = Less than 1 μs.</p> <p>001b = 1 μs – 2 μs.</p> <p>010b = 2 μs – 4 μs.</p> <p>011b = 4 μs – 8 μs.</p> <p>100b = 8 μs – 16 μs.</p> <p>101b = 16 μs – 32 μs.</p> <p>110b = 32 μs – 64 μs.</p> <p>111b = More than 64 μs.</p> <p>All functions share the same value loaded from the NVM. It is reflected in the GLPCI_PMSUP register.</p>
18	RO	0b	Clock Power Management (not supported).
19	RO	0b	Surprise Down Error Reporting Capable. Hardwired to 0b.
20	RO	0b	Data Link Layer Link Active Reporting Capable.
21	RO	0b	Link Bandwidth Notification Capability. Hardwired to 0b (not applicable to endpoints).



Bits	RW	Default	Description
22	HwInit	1b	ASPM Optionality Compliance. Software is permitted to use the value of this bit to help determine whether to enable ASPM or whether to run ASPM compliance tests.
23	RO	0b	Reserved.
31:24	HwInit	0x0	Port Number. The PCIe port number for the given PCIe link. This field is set in the link training phase. The field is hardwired to 0x0.

11.3.5.8 Link Control Register (0xB0; RO)

This register controls PCIe link specific parameters.

Bits	RW	Default	Description
1:0	RW	00b	Active State Link PM Control. This field controls the active state PM enabled on the link. Link PM functionality is determined by the lowest common denominator of all functions. Defined encodings are: 00b = PM Disabled. 01b = L0s entry supported. 10b = L1 entry enabled 11b = L0s and L1 supported. In ARI mode, the ASPM is determined solely by the field in function 0 (even when it is a dummy function) while it is meaningless in the other function(s).
2	RsvdP	0b	Reserved.
3	RW	0b	Read Completion Boundary.
4	RO	0b	Link Disable. Reserved for endpoint devices. Hardwired to 0b.
5	RO	0b	Retrain Link. Not applicable for endpoint devices. Hardwired to 0b.
6	RW	0b	Common Clock Configuration. When set, indicates that the X710-TM4 and the component at the other end of the link are operating with a common reference clock. A value of 0b indicates that they are operating with an asynchronous clock. This parameter affects the L0s exit latencies. In ARI mode, the common clock configuration is determined solely by the field in function 0 (even when it is a dummy function) while it is meaningless in the other function(s).
7	RW	0b	Extended Sync. When set, this bit forces the transmission of additional ordered sets when exiting the L0s state and when in the recovery state. For multi-function devices, if any function has this bit set, then the component must transmit the additional ordered sets when exiting L0s or when in recovery.
8	RO	0b	Reserved.
9	RO/RsvdP	0b	Reserved.
10	RO	0b	Link Bandwidth Management Interrupt Enable. Not applicable to endpoints. Hardwired to 0b.
11	RO	0b	Link Autonomous Bandwidth Interrupt Enable. Not applicable to endpoints. Hardwired to 0b.
15:12	RsvdP	0x0	Reserved.



11.3.5.9 Link Status Register (0xB2; RO)

This register provides information about PCIe link specific parameters.

Bits	RW	Default	Description
3:0	RO	Undefined	Current Link Speed. This field indicates the negotiated link speed of the given PCIe link. The encoded value specifies a bit location in the supported link speeds vector (in the Link Capabilities 2 register) that corresponds to the current link speed. For example, a value of 0010b in this field indicates that the current Link speed is that corresponding to bit 1 in the supported link speeds vector, which is 5.0 GT/s.
9:4	RO	Undefined	Negotiated Link Width. Indicates the negotiated width of the link. Relevant encodings for the X710-TM4 are: 000001b = x1. 000010b = Not supported. 000100b = x4. 001000b = x8. 010000b = Not supported.
10	RO	0b	Undefined.
11	RO	0b	Link Training. Indicates that link training is in progress. This field is not applicable, is reserved for endpoint devices, and is hardwired to 0b.
12	HwInit	1b	Slot Clock Configuration. When set, indicates that the X710-TM4 uses the physical reference clock that the platform provides at the connector. This bit must be cleared if the X710-TM4 uses an independent clock. The <i>Slot Clock Configuration</i> bit is loaded from the NVM. It is reflected in the GLPCI_PMSUP register.
13	RO	0b	Data Link Layer Link Active. Not supported in the X710-TM4. Hardwired to 0b.
14	RO	0b	Link Bandwidth Management Status. Not supported in the X710-TM4. Hardwired to 0b.
15	RO	0b	Link Autonomous Bandwidth Status. This bit is not applicable and is reserved for endpoints.

The following registers are supported only if the capability version is two and above.

11.3.5.10 Device Capabilities 2 Register (0xC4; RO)

This register identifies the PCIe device-specific capabilities.



Bits	RW	Default	Description
3:0	RO	1111b	Completion Timeout Ranges Supported. This field indicates the X710-TM4's support for the optional completion timeout programmability mechanism. Four time value ranges are defined: <ul style="list-style-type: none"> • Range A: 50 μs – 10 ms. • Range B: 10 ms – 250 ms. • Range C: 250 ms – 4 s. • Range D: 4 s – 64 s. Bits are set according to the following values to show the timeout value ranges that the X710-TM4 supports. <ul style="list-style-type: none"> • 0000b = Completion timeout programming not supported. The X710-TM4 must implement a timeout value in the range of 50 μs – 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 values are reserved.
4	RO	1b	Completion Timeout Disable Supported. Note: For dummy functionality, a completion timeout is not relevant as a dummy function because it never sends non-posted requests.
5	RO	0b	ARI Forwarding Supported. Applicable only to Switch Downstream Ports and Root Ports; must be 0b for other function types.
10:6	RO	0x00	Not supported. Hardwired to 0x00.
11	RO	0b	Reserved.
13:12	RO	00b	TPH Completer Supported. Value indicates Completer support for TPH or Extended TPH. This capability is not supported.
15:14	RO	0x0	LN System CLS. Applicable only to Root Ports and RCRBs; must be 00b for all other function types.
17:16	RO	0x0	Reserved.
19:18	HWinit	00b	Reserved.
20	RO		Reserved.
21	RO		Reserved.
23:22	RsvdP		Reserved.
26	RO	0x0	Reserved.
31:24	RsvdP	0x00	Reserved.

11.3.5.11 Device Control 2 Register (0xC8; RW)

This register controls the PCIe specific parameters.



Bits	RW	Default	Description
3:0	RW	0x0	<p>Completion Timeout Value. For devices that support completion timeout programmability, this field enables system software to modify the completion timeout value.</p> <p>Defined encodings:</p> <ul style="list-style-type: none"> • 0000b = Default range: 50 μs to 50 ms. <p>Note: It is strongly recommended that the completion timeout mechanism not expire in less than 10 ms.</p> <p>Values available if Range A (50 μs to 10 ms) programmability range is supported:</p> <ul style="list-style-type: none"> • 0001b = 50 μs to 100 μs. • 0010b = 1 ms to 10 ms. <p>Values available if Range B (10 ms to 250 ms) programmability range is supported:</p> <ul style="list-style-type: none"> • 0101b = 16 ms to 55 ms. • 0110b = 65 ms to 210 ms. <p>Values available if Range C (250 ms to 4 s) programmability range is supported:</p> <ul style="list-style-type: none"> • 1001b = 260 ms to 900 ms. • 1010b = 1 s to 3.5 s. <p>Values available if the Range D (4 s to 64 s) programmability range is supported:</p> <ul style="list-style-type: none"> • 1101b = 4 s to 13 s. • 1110b = 17 s to 64 s. <p>Values not defined are reserved.</p> <p>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. Specifically, FLR clears this field to its default, so that completions are expected to return by the default time.</p> <p>Note: For dummy function, this field is RO - zero.</p>
4	RW	0b	<p>Completion Timeout Disable. When set to 1b, this bit disables the completion timeout mechanism.</p> <p>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.</p> <p>Note: For dummy function, this field is RO - zero.</p>
5	RO	0b	ARI Forwarding Enable. Applicable only to switch devices.
7:6	RO	00b	Not supported. Hardwired to 00b.
8	RW	0b	IDO Request Enable – If this bit is set, the function is permitted to set the ID-Based Ordering (IDO) bit (Attribute[2]) of Requests it initiates.
9	RW	0b	IDO Completion Enable. If this bit is Set, the function is permitted to set the ID-Based Ordering (IDO) bit (Attribute[2]) of Completions it returns.
10	/ RsvdP	0b	Reserved.
12:11	RO	0x0	Reserved.
14:13	RW / RsvdP	00b	<p>OBFF Enable.</p> <p>00b = Disabled</p> <p>01b = Enabled using Message signaling [Variation A]</p> <p>10b = Enabled using Message signaling [Variation B]</p> <p>11b = Enabled using WAKE# signaling</p> <p>For a Multi-Function Device, the field in Function 0 is of type RW, and only Function 0 controls the Component's behavior. In all other Functions of that Device, this field is of type RsvdP.</p>
15	RsvdP	0b	<p>End-End TLP Prefix Blocking.</p> <p>Not applicable to endpoints (RsvdP).</p>



11.3.5.12 Link Capabilities 2 Register (0xCC)

Bits	RW	Default	Description
0	RsvdP	0b	Reserved.
7:1	RO	0x01	Supported Link Speeds Vector. This field indicates the supported Link speed(s) of the associated Port. For each bit, a value of 1b indicates that the corresponding link speed is supported; otherwise, the Link speed is not supported. Bit definitions are: Bit 1 = 2.5 GT/s. Bit 2 = 5.0 GT/s. Bit 3 = 8.0 GT/s. Bit 4 = Not supported. Bits 7:5 = RsvdP. Multi-function devices associated with the same Upstream Port must report the same value in this field for all functions. This field is loaded from the NVM and reflected in the GLPCLINKCAP register.
8	RO	0b	Crosslink Supported. When set to 1b, this bit indicates that the associated Port supports crosslinks. It is recommended that this bit be set in any port that supports crosslinks even though doing so is only required for ports that also support operating at 8.0 GT/s or higher Link speeds.
31:9	RsvdP	0x00	Reserved.

11.3.5.13 Link Control 2 Register (0xD0; RWS)

Bits	RW	Default	Description
3:0	RWS (func 0) / RsvdP (else)	0011b (func 0) 0000b (else)	Target Link Speed. This field is used to set the target compliance mode speed when software is using the <i>Enter Compliance</i> bit to force a link into compliance mode. The encoding is the binary value of the bit in the Supported Link Speeds Vector (in the Link Capabilities 2 register) that corresponds to the desired target Link speed. All other encodings are Reserved. For example, 5.0 GT/s corresponds to bit 1 in the Supported Link Speeds Vector, so the encoding for a 5.0 GT/s target Link speed in this field is 0010b If a value is written to this field that does not correspond to a speed included in the <i>Supported Link Speeds</i> field, the result is undefined. The default value of this field is the highest link speed supported by the X710-TM4 (as reported in the <i>Supported Link Speeds</i> field of the Link Capabilities register).
4	RWS (func 0) / RsvdP (else)	0b	Enter Compliance. Software is permitted to force a link to enter compliance mode at the speed indicated in the <i>Target Link Speed</i> 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.
5	RWS (func 0) / RsvdP (else)	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.
6	RO	0b	Selectable De-Emphasis. This bit is not applicable and reserved for endpoints.



Bits	RW	Default	Description
9:7	RWS (func 0) / RsvdP (else)	000b	Transmit Margin. This field controls the value of the non de emphasized voltage level at the Transmitter pins. Encodings: 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.
10	RWS (func 0) / RsvdP (else)	0b	Enter Modified Compliance. When this bit is set to 1b, the device transmits modified compliance pattern if the LTSSM enters Polling.Compliance state. The default value of this bit is 0b.
11	RWS (func 0) / RsvdP (else)	0b	Compliance SOS. When set to 1b, the LTSSM is required to send SOS periodically in between the (modified) compliance patterns. This bit is applicable when the Link is operating at 2.5 GT/s or 5 GT/s data rates only. The default value of this bit is 0b.
15:12	RWS (func 0) / RsvdP (else)	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, the setting of this bit field has no effect. Defined Encodings are: 0001b = 3.5 dB. 0000b = 6 dB. For a multi-function device associated with an Upstream Port, the bit field in Function 0 is of type RWS, and only Function 0 controls the component's Link behavior. In all other Functions of that device, this bit field is of type RsvdP. This bit field is intended for debug, and compliance testing purposes. The default value of this field is 0000b.

11.3.5.14 Link Status 2 Register (0xD2; RW)

Bits	RW	Default	Description
0	RO	0b	Current De-emphasis Level. When the link is operating at 5 GT/s speed, this bit reflects the level of de-emphasis. it is undefined when the Link is not operating at 5.0 GT/s speed. Encodings: 1b = 3.5 dB. 0b = 6 dB. Note: Same value must be reported for all functions.
1	ROS/RsvdZ	0b	Equalization Complete. When set to 1b, this bit indicates that the Transmitter Equalization procedure completed. Note: This bit must be implemented in Function 0 and RsvdZ in other Functions.
2	ROS/RsvdZ	0b	Equalization Phase 1 Successful. When set to 1b, this bit indicates that Phase 1 of the Transmitter Equalization procedure has successfully completed. Note: This bit must be implemented in Function 0 and RsvdZ in other Functions.
3	ROS/RsvdZ	0b	Equalization Phase 2 Successful. When set to 1b, this bit indicates that Phase 2 of the Transmitter Equalization procedure has successfully completed. Note: This bit must be implemented in Function 0 and RsvdZ in other Functions.



Bits	RW	Default	Description
4	ROS/RsvdZ	0b	Equalization Phase 3 Successful. When set to 1b, this bit indicates that Phase 3 of the Transmitter Equalization procedure has successfully completed. Note: This bit must be implemented in Function 0 and RsvdZ in other Functions.
5	RW1C/ RsvdZ	0b	Link Equalization Request. This bit is set by hardware to request the link equalization process to be performed on the link. Note: This bit must be implemented in Function 0 and RsvdZ in other functions.
15:6	RsvdZ	0x00	Reserved.

11.4 PCIe Extended Configuration Space

PCIe configuration space is located in a flat memory-mapped address space. PCIe extends the configuration space beyond the 256 bytes available for PCI to 4096 bytes. The X710-TM4 decodes an additional four bits (bits 27:24) to provide the additional configuration space as shown. PCIe reserves the remaining four bits (bits 31:28) for future expansion of the configuration space beyond 4096 bytes.

The configuration address for a PCIe device is computed using a PCI-compatible bus, device, and function numbers as follows:

31	28	27	20	19	15	14	12	11	2	1	0
0000b		Bus #		Device #		Fun #		Register Address (offset)		00b	

PCIe extended configuration space is allocated using a linked list of optional or required PCIe extended capabilities following a format resembling PCI capability structures. The first PCIe extended capability is located at offset 0x100 in the device configuration space. The first Dword of the capability structure identifies the capability/version and points to the next capability.

The X710-TM4 supports the following PCIe extended capabilities:

Table 11-13. Extended Capabilities List

Address range	Item	Cases Where Capability Does Not Exist	Next Pointer
0x100 - 0x128	Advanced Error Reporting (AER)	None (always present)	Any of the below / 0x000
0x140 - 0x148	Device Serial Number	NVM is not valid	Any of the below / 0x000
0x150 - 0x154	Alternative RID Interpretation (ARI)	ARI Enabled bit in the NVM is set to 0b	Any of the below / 0x000
0x160 - 0x19C	Single Root I/O Virtualization (SR-IOV)	<ul style="list-style-type: none"> The global SR-IOV Enable bit in NVM is set to 0b (exposed via the <i>GLPCI_CAPSUP.IOV_EN</i> bit). This is a dummy function. The per-PF SR-IOV Enable bit is set to 0b (<i>PF_VT_PFALLOC.VALID</i> is cleared). 	Any of the below / 0x000



Table 11-13. Extended Capabilities List

Address range	Item	Cases Where Capability Does Not Exist	Next Pointer
0x1A0 - 0x1A8	TPH Requester	<ul style="list-style-type: none"> TPH Enabled bit in NVM is set to 0b. This is a dummy function. 	Any of the below / 0x000
0x1B0 - 0x1B4	Access Control Services (ACS)	<ul style="list-style-type: none"> ACS Enabled bit in NVM is set to 0b. A single PF is enabled and SR-IOV is disabled. 	Any of the below / 0x000
0x1D0 - 0x1E8	Secondary PCI Express	<ul style="list-style-type: none"> Secondary PCIe Enabled bit in NVM is set to 0b. Function is not function 0. 	0x000

11.4.1 Advanced Error Reporting Capability (AER)

The PCIe advanced error reporting capability is an optional extended capability to support advanced error reporting. The tables that follow list the PCIe advanced error reporting extended capability structure for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x100	Next Capability Ptr.		Version (0x1)	AER Capability ID (0x0001)
0x104	Uncorrectable Error Status			
0x108	Uncorrectable Error Mask			
0x10C	Uncorrectable Error Severity			
0x110	Correctable Error Status			
0x114	Correctable Error Mask			
0x118	Advanced Error Capabilities and Control Register			
0x11C... 0x128	Header Log			

Table 11-14 lists configuration sharing of the AER Capability registers among the different PCI functions.

Table 11-14. Configuration sharing of the AER Capability

Field	Sub-field	Shared?	Replicated?	Comments
Enhanced Capability Header Register	Extended Capability ID	x		
	Capability Version	x		
	Next Capability Offset		x	
Uncorrectable Error Status			x	



Table 11-14. Configuration sharing of the AER Capability

Field	Sub-field	Shared?	Replicated?	Comments
Uncorrectable Error Mask			x	
Uncorrectable Error Severity			x	
Correctable Error Status			x	
Correctable Error Mask			x	
Advanced Error Capabilities and Control	First Error Pointer		x	
	ECRC Generation Capable	x		
	ECRC Generation Enable		x	ECRC insertion is per PF.
	ECRC Check Capable	x		
	ECRC Check Enable		x	See Section 3.1.2.9.
Header Log			x	

11.4.1.1 Advanced Error Reporting Enhanced Capability Header Register (0x100; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x0001	Extended Capability ID. PCIe extended capability ID indicating advanced error reporting capability.
19:16	RO	0x2	Version Number. PCIe advanced error reporting extended capability version number.
31:20	RO	See Description	Next Capability Offset. Next PCIe extended capability offset. See Table 11-13 for possible values of the next capability offset.

11.4.1.2 Uncorrectable Error Status Register (0x104; RW1CS)

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.

Bit Location	Attribute	Default Value	Description
0	RO	0b	Reserved.
3:1	RsvdZ	000b	Reserved.
4	RW1CS	0b	Data Link Protocol Error Status.



Bit Location	Attribute	Default Value	Description
5	RO	0b	Reserved.
11:6	RsvdZ	0x00	Reserved.
12	RW1CS	0b	Poisoned TLP Received Status.
13	RW1CS	0b	Flow Control Protocol Error Status.
14	RW1CS	0b	Completion Timeout Status.
15	RW1CS	0b	Completer Abort Status.
16	RW1CS	0b	Unexpected Completion Status.
17	RW1CS	0b	Receiver Overflow Status.
18	RW1CS	0b	Malformed TLP Status.
19	RW1CS	0b	ECRC Error Status.
20	RW1CS	0b	Unsupported Request Error Status.
21	RO	0b	ACS Violation Status. Not supported. Hardwired to 0b.
25:22	RO	0x0	Not supported.
31:26	RsvdP	0b	Reserved.

11.4.1.3 Uncorrectable Error Mask Register (0x108; RWS)

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.

Bit Location	Attribute	Default Value	Description
0	RO	0b	Reserved.
3:1	RsvdP	000b	Reserved.
4	RWS	0b	Data Link Protocol Error Mask.
5	RO	0b	Reserved.
11:6	RsvdP	0x00	Reserved.
12	RWS	0b	Poisoned TLP Received Mask.
13	RWS	0b	Flow Control Protocol Error Mask.
14	RWS	0b	Completion Timeout Mask.
15	RWS	0b	Completer Abort Mask.
16	RWS	0b	Unexpected Completion Mask.
17	RWS	0b	Receiver Overflow Mask.
18	RWS	0b	Malformed TLP Mask.
19	RWS	0b	ECRC Error Mask.
20	RWS	0b	Unsupported Request Error Mask.



Bit Location	Attribute	Default Value	Description
21	RO	0b	ACS Violation Mask. Not supported. Hardwired to 0b.
25:22	RO	0x0	Not supported
31:26	RsvdP	0b	Reserved

11.4.1.4 Uncorrectable Error Severity Register (0x10C; RWS)

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.

Bit Location	Attribute	Default Value	Description
0	RO	0b	Reserved.
3:1	RsvdP	000b	Reserved.
4	RWS	1b	Data Link Protocol Error Severity.
5	RO	0b	Reserved.
11:6	RsvdP	0x00	Reserved.
12	RWS	0b	Poisoned TLP Received Severity.
13	RWS	1b	Flow Control Protocol Error Severity.
14	RWS	0b	Completion Timeout Severity.
15	RWS	0b	Completer Abort Severity.
16	RWS	0b	Unexpected Completion Severity.
17	RWS	1b	Receiver Overflow Severity.
18	RWS	1b	Malformed TLP Severity.
19	RWS	0b	ECRC Error Severity.
20	RWS	0b	Unsupported Request Error Severity.
21	RO	0b	ACS Violation Severity. Not supported. Hardwired to 0b.
25:22	RO	0x0	Not supported.
31:26	RsvdP	0b	Reserved.

11.4.1.5 Correctable Error Status Register (0x110; RW1CS)

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.



Bit Location	Attribute	Default Value	Description
0	RW1CS	0b	Receiver Error Status.
5:1	RsvdZ	0b	Reserved.
6	RW1CS	0b	Bad TLP Status.
7	RW1CS	0b	Bad DLLP Status.
8	RW1CS	0b	REPLAY_NUM Rollover Status.
11:9	RsvdZ	0b	Reserved.
12	RW1CS	0b	Replay Timer Timeout Status.
13	RW1CS	0b	Advisory Non-Fatal Error Status.
15:14	RO	0b	Reserved.
31:16	RsvdZ	0x00	Reserved.

11.4.1.6 Correctable Error Mask Register (0x114; RWS)

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.

Bit Location	Attribute	Default Value	Description
0	RWS	0b	Receiver Error Mask.
5:1	RsvdP	0b	Reserved.
6	RWS	0b	Bad TLP Mask.
7	RWS	0b	Bad DLLP Mask.
8	RWS	0b	REPLAY_NUM Rollover Mask.
11:9	RsvdP	0b	Reserved.
12	RWS	0b	Replay Timer Timeout Mask.
13	RWS	1b	Advisory Non-Fatal Error Mask.
15:14	RO	0b	Reserved.

11.4.1.7 Advanced Error Capabilities and Control Register (0x118; RO)

Bit Location	Attribute	Default Value	Description
4:0	ROS	0b	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.



Bit Location	Attribute	Default Value	Description
5	RO	1b	ECRC Generation Capable. If set, this bit indicates that the function is capable of generating ECRC. This bit is loaded from NVM. It is reflected in the GLPCI_CAPSUP register.
6	RWS	0b	ECRC Generation Enable. When set, ECRC generation is enabled.
7	RO	1b	ECRC Check Capable. If set, this bit indicates that the function is capable of checking ECRC. This bit is loaded from NVM. It is reflected in the GLPCI_CAPSUP register.
8	RWS	0b	ECRC Check Enable. When set Set, ECRC checking is enabled.
9	RO	0b	Multiple Header Recording Capable. Not supported. Hardwired to 0b.
10	RO	0b	Multiple Header Recording Enable. Not supported. Hardwired to 0b.
11	RsvdP	0b	TLP Prefix Log Present. Not supported. Hardwired to 0b.
15:12	RsvdP	0x0	Reserved.

11.4.1.8 Header Log Register (0x11C:0x128; RO)

The header log register captures the header for the transaction that generated an error. This register is 16 bytes.

Bit Location	Attribute	Default Value	Description
127:0	ROS	0b	Header of the packet in error (TLP or DLLP).

11.4.2 Serial Number

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.

Serial Number capability is implemented for function 0; all other functions return the same device serial number value as that reported by function 0.

The capability is disabled when the MAC address in the GLPCI_SERL and GLPCI_SERH registers is 0x00...0 (indicating that the NVM is not valid or a proper MAC address was not loaded).

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x140	Next Capability Ptr.		Version (0x1)	Serial ID Capability ID (0x0003)
0x144	Serial Number Register (Lower Dword)			
0x148	Serial Number Register (Upper Dword)			



Table 11-15 lists configuration sharing of the Serial Number Capability registers among the different PCI functions.

Table 11-15. Configuration Sharing of the Serial Number Capability

Field	Sub-field	Shared?	Replicated?	Comments
Enhanced Capability Header Register	Extended Capability ID	x		
	Capability Version	x		
	Next Capability Offset		x	
Serial Number Register		x		See previous comment.

11.4.2.1 Device Serial Number Enhanced Capability Header Register (0x140; RO)

Bit(s)	Attribute	Default Value	Description
15:0	RO	0x0003	PCIe Extended Capability ID. This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. The extended capability ID for the device serial number capability is 0x0003.
19:16	RO	0x1	Capability Version. This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Note: Must be set to 0x1 for this version of the specification.
31:20	RO	See description	Next Capability Offset. This field contains the offset to the next PCIe capability structure or 0x000 if no other items exist in the linked list of capabilities. See Table 11-13 for possible values of the next capability offset.

11.4.2.2 Serial Number Registers (0x144:0x148; RO)

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit Extended Unique Identifier (EUI-64*). The register at offset 0x144 holds the lower 32 bits and the register at offset 0x148 holds the higher 32 bits. The following figure details the allocation of register fields in the Serial Number register. The table that follows lists the respective bit definitions.

Bit(s)	Attributes	Description
63:0	RO	PCIe Device Serial Number. This field contains the IEEE defined 64-bit EUI-64*. This identifier includes a 24-bit company ID value assigned by IEEE registration authority and a 40-bit extension identifier assigned by the manufacturer.

The serial number uses the Ethernet MAC address according to the following definition:



Field	Company ID			Extension Identifier				
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7
Most Significant Byte						Least Significant Byte		
Most Significant Bit						Least Significant Bit		

The serial number can be constructed from the 48-bit Ethernet MAC address in the following form:

Field	Company ID			MAC Label		Extension identifier		
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7
Most Significant Bytes						Least Significant Byte		
Most Significant Bit						Least Significant Bit		

In this case, the MAC label is 0xFFFF.

For example, assume that the company ID is (Intel) 00-A0-C9 and the extension identifier is 23-45-67 (MAC address of 00-A0-C9-23-45-67). In this case, the 64-bit serial number is:

Field	Company ID			MAC Label		Extension Identifier		
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7
	00	A0	C9	FF	FF	23	45	67
Most Significant Byte						Least Significant Byte		
Most Significant Bit						Least Significant Bit		

The Ethernet MAC address for the serial number capability is loaded from NVM (not the same field that is loaded from NVM into the Station MAC Address registers). It is reflected in the GLPCI_SERL and GLPCI_SERH registers. In the above example:

- GLPCI_SERL = C9-23-45-67
- GLPCI_SERH = 00-00-00-A0

Note: The official document that defines EUI-64* is: <http://standards.ieee.org/regauth/oui/tutorials/EUI64.html>

11.4.3 Alternate Routing ID Interpretation (ARI) Capability Structure

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. The capability is exposed when the GLPCI_CAPSUP.ARI_EN bit is set from NVM.



The ARI capability structure is as follows:

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x150	Next Capability Ptr.		Version (0x1)	ARI Capability ID (0x000E)
0x154	ARI Control Register		ARI Capability Register	

Table 11-16 lists configuration sharing of the ARI Capability registers among the different PCI functions.

Table 11-16. Configuration Sharing of the ARI Capability

Field	Sub-field	Shared?	Replicated?	Comments
Enhanced Capability Header Register	Extended Capability ID	x		
	Capability Version	x		
	Next Capability Offset		x	
ARI Capability Register	Next Function Pointer		x	

11.4.3.1 PCIe ARI Header Register (0x150; RO)

Field	Bit(s)	Initial Value	Access	Description
ID	15:0	0x000E	RO	PCIe Extended Capability ID. PCIe extended capability ID for the alternative RID interpretation.
Version	19:16	1b	RO	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.
Next Capability Offset	31:20	See Description	RO	Next Capability Offset. This field contains the offset to the next PCIe extended capability structure. See Table 11-13 for possible values of the next capability offset.

11.4.3.2 PCIe ARI Capability Register (0x154; RO)

Field	Bit(s)	Initial Value	Access	Description
M	0	0b	RO	Reserved.
A	1	0b	RO	Reserved.
Reserved	7:2	0b	RsvdP	Reserved.



Field	Bit(s)	Initial Value	Access	Description
NFN	15:8	See Description ¹	RO	Next Function Number. This field contains the pointer to the next PF configuration space or 0x0000 if no other items exist in the linked list of functions. Function 0 is the start of the link list of functions. Functions might be disabled during the POR flow (through strapping pins, SMASH/CLP commands, NC-SI commands) affecting this field.
M_EN	16	0b	RO	Reserved.
A_EN	17	0b	RO	Reserved.
Reserved	19:18	00b	RO	Reserved.
FGN	22:20	0b	RO	Reserved.
Reserved	31:23	0b	RsvdP	Reserved.

1. If function zero is a dummy function, this register should keep its attributes according to the function number. Disabled functions are skipped.

11.4.4 SR-IOV Capability Structure

This is a structure used to support the SR-IOV capabilities reporting and control. The capability is exposed when the GLPCI_CAPSUP.IOV_EN bit is set from NVM and the PF_VT_PFALLOC.VALID is set.

The following tables shows the implementation of this structure in the X710-TM4.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x160	Next Capability Offset		Version (0x1)	SR-IOV Capability ID (0x0010)
0x164	SR-IOV Capabilities			
0x168	SR-IOV Status		SR-IOV Control	
0x16C	TotalVFs (RO)		Initial VF (RO)	
0x170	Reserved	Function Dependency Link (RO)	Num VF (RW)	
0x174	VF Stride (RO)		First VF Offset (RO)	
0x178	VF Device ID		Reserved	
0x17C	Supported Page Size (0x553)			
0x180	system page Size (RW)			
0x184	VF BAR0 – Low (RW)			
0x188	VF BAR0 – High (RW)			
0x18C	VF BAR2 (RO)			
0x190	VF BAR3 – Low (RW)			
0x194	VF BAR3- High (RW)			
0x198	VF BAR5 (RO)			
0x19C	VF Migration State Array Offset (RO)			



Table 11-17 lists configuration sharing of the SR-IOV Capability registers among the different PCI functions.

Table 11-17. Configuration Sharing of the SR-IOV Capability

Field	Sub-field	Shared?	Replicated?	Comments
Enhanced Capability Header Register	Extended Capability ID	x		
	Capability Version	x		
	Next Capability Offset		x	
SR-IOV Capabilities	VF Migration Capable	x		Not supported.
	ARI Capable Hierarchy Preserved		x	PF0 only. RO zero in all other functions.
	VF Migration Interrupt Message Number			Not supported.
SR-IOV Control	VF Enable		x	
	Memory Space Enable		x	
	ARI Capable Hierarchy	x		PF0 only. RO zero in all other functions.
InitialVFs			x	
TotalVFs			x	
Num VFs			x	
Function Dependency Link			x	Each PF indicates its PF number here.
First VF Offset			x	
VF Stride			x	
VF Device ID			x	
Supported Page Size		x		
System Page Size			x	
VF BARs			x	

11.4.4.1 PCIe SR-IOV Header Register (0x160; RO)

Field	Bit(s)	Initial Value	Access	Description
ID	15:0	0x0010	RO	PCIe Extended Capability ID. PCIe extended capability ID for the SR-IOV capability.
Version	19:16	0x1	RO	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.
Next pointer	31:20	0x0	RO	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. See Table 11-13 for possible values of the next capability offset.



11.4.4.2 PCIe SR-IOV Capabilities Register (0x164; RO)

Field	Bit(s)	Initial Value	Access	Description
VFMC	0	0b	RO	Reserved.
ARI CHP	1	1b (lowest SR-IOV-enabled function) / 0b (else)	RO	ARI Capable Hierarchy Preserved. If set, the ARI Capable Hierarchy bit is preserved across certain power state transitions. Only present in lowest SR-IOV-enabled function. Read Only Zero in other PFs.
Reserved	20:2	0x0	RO	Reserved.
IMN	31:21	0x0	RO	Reserved.

11.4.4.3 PCIe SR-IOV Control Register (0x168; RW)

Field	Bit(s)	Initial Value	Access	Description
VFE	0	0b	RW	<p>VF Enable.</p> <p>VF Enable manages the assignment of VFs to the associated PF. If <i>VF Enable</i> is set to 1b, 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.</p> <p>If set to 0b, VFs must be disabled and not visible in the PCIe fabric; VFs cannot respond to or issue PCIe transactions.</p> <p>In addition, if <i>VF Enable</i> is cleared after having been set, all of the VFs must no longer:</p> <ul style="list-style-type: none"> • Issue PCIe transactions • Respond to configuration space or memory space accesses. <p>The behavior must be as if an FLR was issued to each of the VFs. Specifically, VFs must not retain any context after <i>VF Enable</i> 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.</p>
VF ME	1	0b	RO	Reserved.
VF MIE	2	0b	RO	Reserved.
VF MSE	3	0b	RW	<p>Memory Space Enable for Virtual Functions.</p> <p>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.</p> <p>When VF Enable is 1, virtual function memory space access is permitted only when VF MSE is Set. VFs must follow the same error reporting rules as defined in the base specification if an attempt is made to access a VFs memory space when VF Enable is 1b and VF MSE is 0b.</p> <p>Implementation Note: VFs memory space cannot be accessed when VF Enable is zero. Thus, VF MSE is don't care when VF Enable is zero; however, software might choose to set VF MSE after programming the VF BARn registers, prior to setting VF Enable to 1b.</p>



Field	Bit(s)	Initial Value	Access	Description
VF ARI	4	0b	RW (lowest SR-IOV-enabled function) RO (else)	ARI Capable Hierarchy. Device can locate VFs in function numbers 8 to 255 of the captured bus number. If either ARI Capable Hierarchy Preserved is Set or No_Soft_Reset is Set, a power state transition of this PF from D3hot to D0 does not affect the value of this bit.
Reserved	15:5	0x0	RO	Reserved.
VFMIS	16	0b	RO	Reserved.
Reserved	31:17	0b	RO	Reserved.

11.4.4.4 PCIe SR-IOV Initial/Total VFs Register (0x16C; RO)

Field	Bit(s)	Initial Value	Access	Description
InitialVFs	15:0	See Section 1.5.1.1	RO	InitialVFs indicates the number of VFs that are initially associated with the PF. If <i>VF Migration Capable</i> is cleared, this field must contain the same value as TotalVFs. In the X710-TM4 this parameter is equal to the TotalVFs in this register.
TotalVFs	31:16	See Section 1.5.1.1	RO	TotalVFs defines the maximum number of VFs that can be associated with the PF. This field is derived from the PF_VT_PFALLOC.FIRSTVF and PF_VT_PFALLOC register fields loaded from the NVM.

11.4.4.5 PCIe SR-IOV Num VFs Register (0x170; RW)

Field	Bit(s)	Initial Value	Access	Description
NumVFs	15:0	0x0	RW	Num VFs 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 <i>VF Enable</i> is set to 1b.
FDL	23:16	0x0 (func 0) ¹ 0x1 (func 1) ... 0xn (func n) ...	RO	Function Dependency Link. Defines dependencies between physical functions allocation. In the X710-TM4 there are no constraints.
Reserved	31:24	0	RO	Reserved.

1. Applies to dummy function as well.



11.4.4.6 PCIe SR-IOV VF RID Mapping Register (0x174; RO)

Field	Bit(s)	Initial Value	Access	Description
FVO	15:0	0x100 + 0x10 + FirstVF - PF#	RO	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 first VFs 16-bit RID is calculated by adding the contents of this field to the RID of the PF containing this field. The content of this field is valid only when <i>VF Enable</i> is set. If <i>VF Enable</i> is 0b, the contents are undefined. If the <i>VF ARI</i> bit is set, this field changes to 0x10 + FirstVF - PF#. This field is derived from the PF_VT_PFALLOC.FIRSTVTF register field loaded from NVM
VFS	31:16	0x1 ¹	RO	VF stride defines the 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 <i>VF Enable</i> is set and <i>NumVFs</i> is non-zero. If <i>VF Enable</i> is 0b or if <i>NumVFs</i> is zero, the contents are undefined.

1. See Section 11.5.1.1.

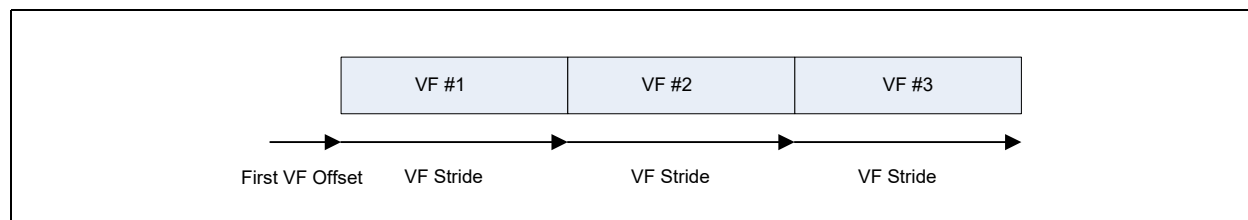


Figure 11-1. VF Stride

11.4.4.7 PCIe SR-IOV VF Device ID Register (0x178; RO)

All VFs have the same default value of 0x154C, and can be auto-loaded from the NVM.

The VF Device ID is loaded from the NVM according to the following rules:

- Device ID is loaded from the NVM if the GLPCI_CAPSUP.LOAD_DEV_ID bit is set.
- The Device ID value of all VFs associated with a given PF is loaded to the respective *PFPCI_DEVID.VF_DEV_ID* field.



11.4.4.8 PCIe SR-IOV Supported Page Size Register (0x17C; RO)

Field	Bit(s)	Initial Value	Access	Description
Supported Page Size	31:0	0x553	RO	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 4KB 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.

11.4.4.9 PCIe SR-IOV System Page Size Register (0x180; RW)

Field	Bit(s)	Initial Value	Access	Description
Page Size	31:0	0x1	RW	This field defines the page size the system uses to map the VFs' memory addresses. Software must set the value of the <i>System Page Size</i> to one of the page sizes set in the <i>Supported Page Sizes</i> field. As with <i>Supported Page Sizes</i> , if bit n is set in <i>System Page Size</i> , the 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 <i>System Page Size</i> . The results are undefined if a bit is set in <i>System Page Size</i> that is not set in <i>Supported Page Sizes</i> . When <i>System Page Size</i> is set, the VFs are required to align all BAR resources on a <i>System Page Size</i> boundary. Each BAR size, including <i>VF BARn Size</i> (described later) must be aligned on a <i>System Page Size</i> boundary. Each BAR size, including <i>VF BARn Size</i> must be sized to consume a multiple of <i>System Page Size</i> bytes. All fields requiring page size alignment within a function must be aligned on a <i>System Page Size</i> boundary. <i>VF Enable</i> must be zero when <i>System Page Size</i> is set. The results are undefined if <i>System Page Size</i> is set when <i>VF Enable</i> is set.

11.4.4.10 PCIe SR-IOV BAR 0 – Low Register (0x184; RW)

Field	Bit(s)	Initial Value	Access	Description
Mem	0	0b	RO	0b indicates memory space.
Mem Type	2:1	10b	RO	Indicates the address space size. 10b = 64-bit. This bit is loaded from the NVM. It is reflected in the GLPCI_VFSUP register
Prefetch Mem	3	0b	RO	0b = Non-prefetchable space. 1b = Prefetchable space. This bit is loaded from the NVM. It is reflected in the GLPCI_VFSUP register



Field	Bit(s)	Initial Value	Access	Description
Memory Address Space	31:4	0x0	RW	Which bits are RW bits and which are RO to 0x0 depend on the memory mapping window size.

11.4.4.11 PCIe SR-IOV BAR 0 – High Register (0x188; RW)

Field	Bit(s)	Initial Value	Access	Description
BAR0 – MSB	31:0	0x0	RW	MSB part of BAR0.

11.4.4.12 PCIe SR-IOV BAR 2 Register (0x18C; RO)

Field	Bit(s)	Initial Value	Access	Description
BAR2	31:0	0x0	RO	This BAR is not used.

11.4.4.13 PCIe SR-IOV BAR 3 – Low Register (0x190; RW)

Field	Bit(s)	Initial Value	Access	Description
Mem	0	0b	RO	0b indicates memory space.
Mem Type	2:1	10b	RO	Indicates the address space size. 10b = 64-bit. This bit is loaded from the NVM. It is reflected in the GLPCI_VFSUP register
Prefetch Mem	3	0b*	RO	0b = Non-prefetchable space 1b = Prefetchable space This bit is loaded from the NVM. It is reflected in the GLPCI_VFSUP register
Memory Address Space	31:4	0x0	RW	Which bits are RW bits and which are RO to 0x0 depend on the memory mapping window size. The size is a maximum between 16 KB and page size.



11.4.4.14 PCIe SR-IOV BAR 3 – High Register (0x194; RW)

Field	Bit(s)	Initial Value	Access	Description
BAR3 – MSB	31:0	0x0	RW	MSB part of BAR3.

11.4.4.15 PCIe SR-IOV BAR 5 Register (0x198; RO)

Field	Bit(s)	Initial Value	Access	Description
BAR5	31:0	0x0	RO	This BAR is not used.

11.4.4.16 PCIe SR-IOV VF Migration State Array Offset Register (0x19C; RO)

Field	Bit(s)	Initial Value	Access	Description
BIR	2:0	0x0	RO	Reserved.
Offset	31:0	0x0	RO	Reserved.

11.4.5 TPH Requester Capability

The TPH Requester capability is an optional extended capability to support TLP Processing Hints. The capability is exposed when the GLPCI_CAPSUP.TPH_EN bit is set from the NVM.

The following table lists the TPH extended capability structure for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x1A0	PCIe Extended Capability Header			
0x1A4	TPH Requester Capability Register			
0x1A8	TPH Requester Control Register			

Table 11-18 lists configuration sharing of the TPH Requester Capability registers among the different PCI functions.



Table 11-18. Configuration Sharing of the TPH Requester Capability

Field	Sub-field	Shared?	Replicated?	Comments
Enhanced Capability Header Register	Extended Capability ID	x		
	Capability Version	x		
	Next Capability Offset		x	
TPH Requester Capability		x		
TPH Requester Control			x	
TPH ST Table			x	The Steering Table Upper fields are not supported.

11.4.5.1 TPH Requester Extended Capability Header (0x1A0; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x17	Extended Capability ID. PCIe extended capability ID indicating TPH capability.
19:16	RO	0x1	Capability Version. PCIe TPH extended capability version number.
31:20	RO	See Desc.	Next Capability Offset. This field contains the offset to the next PCIe capability structure. See Table 11-13 for possible values of the next capability offset.

11.4.5.2 TPH Requester Capability Register (0x1A4; RO)

Bit Location	Attribute	Default Value	Description
0	RO	1	No ST Mode Supported. If set indicates that the Function supports the No ST Mode of operation.
1	RO	0	Reserved.
2	RO	1	Device Specific Mode. Set to indicate that the X710-TM4 supports Device Specific Mode of operation.
7:3	RsvdP	0	Reserved.
8	RO	0	Extended TPH Requester Supported. Cleared to indicate that the function is not capable of generating requests with Extended TPH TLP Prefix.
10:9	RO	00b	ST Table Location. Value indicates if and where the ST Table is located. Defined encodings are: 00b = ST Table is not present. 01b = ST Table is located in the TPH Requester Capability structure. 10b = ST Table is located in the MSI-X Table structure. 11b = Reserved. ST Table is not supported.



Bit Location	Attribute	Default Value	Description
15:11	RsvdP	0x0	Reserved.
26:16	RO	0x0	ST_Table Size. System software reads this field to determine the ST_Table_Size N, which is encoded as N-1. For example, a returned value of 0000000011. indicates a table size of 4. The value in this field is undefined since the X710-TM4 does not support an ST Table
31:27	RsvdP	0x0	Reserved.

11.4.5.3 TPH Requester Control Register (0x1A8; R/W)

Bit Location	Attribute	Default Value	Description
2:0	RW	0x0	ST Mode Select. Indicates the ST mode of operation selected. Defined encodings are: 000b = No table mode. 001b = Reserved. 010b = Device specific mode. Others = Reserved for future use. The default value of 000 indicates No Table mode of operation.
7:3	RsvdP	0x0	Reserved
9:8	RW	0x0	TPH Requester Enable. Controls the ability to issue Request TLPs using either TPH or Extended TPH. Defined encodings are: 00b = The X710-TM4 is not permitted to issue transactions with TPH or Extended TPH as Requester. 01b = The X710-TM4 is permitted to issue transactions with TPH as Requester and is not permitted to issue transactions with Extended TPH as Requester. 10b = Reserved. 11b = The X710-TM4 is permitted to issue transactions with TPH and Extended TPH as Requester (the X710-TM4 does not issue transactions with Extended TPH). The default value of this field is 00b.
31:10	RsvdP	0x0	Reserved.

11.4.6 ACS Extended Capability Structure

The ACS Extended Capability defines a set of control points within a PCIe topology to determine whether a TLP should be routed normally, blocked, or redirected. The capability is exposed when the GLPCI_CAPSUP.ACS_EN bit is set from the NVM.

The ACS Capability structure is shared and exposed to all PFs.

The following table lists the PCIe ACS extended capability structure for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x1B0	PCIe Extended Capability Header			
0x1B4	ACS Control Register (0x0)		ACS Capability Register (0x0)	



11.4.6.1 ACS Extended Capability Header (0x1B0; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x0D	PCIe Extended Capability ID. PCIe extended capability ID indicating ACS capability.
19:16	RO	0x1	Capability Version. PCIe ACS extended capability version number.
31:20	RO	See Desc.	Next Capability Offset. See Table 11-13 for possible values of the next capability offset.

11.4.6.2 ACS Capability Register (0x1B4; RO)

Bit Location	Attribute	Default Value	Description
0	RO	0b	Reserved.
1	RO	0b	Reserved.
2	RO	0b	Reserved.
3	RO	0b	Reserved.
4	RO	0b	Reserved.
5	RO	0b	Reserved.
6	RO	0b	Reserved.
7	RsrvP	0b	Reserved.
15:8	RO	0x0	Reserved.

11.4.6.3 ACS Control Register (0x1B6; RO)

Bit Location	Attribute	Default Value	Description
0	RO	0b	Reserved.
1	RO	0b	Reserved.
2	RO	0b	Reserved.
3	RO	0b	Reserved.
4	RO	0b	Reserved.
5	RO	0b	Reserved.
6	RO	0b	Reserved.
15:7	RsvdP	0b	Reserved.



11.4.7 Secondary PCIe Extended Capability

The Secondary PCIe 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. The capability is exposed when the GLPCI_CAPSUP.SEC_EN bit is set from NVM.

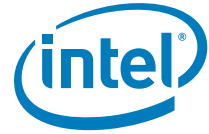
The following table lists the Secondary PCI Express extended capability structure for PCIe devices.

11.4.7.1 Secondary PCIe Extended Capability

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x1D0	PCI Express Extended Capability Header			
0x1D4	Link Control 3 Register			
0x1D8	Lane Error Status Register			
0x1DC	Equalization Control Register (Sized by Maximum Link Width)			
...	...			

11.4.7.2 Header (0x1D0)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x19	PCIe Extended Capability ID. This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability. PCIe Extended Capability ID for the Secondary PCIe Extended Capability is 0x0019.
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 1h for this version of the specification.
31:20	RO	See Desc.	Next Capability Offset. See Table 11-13 for possible values of the next capability offset.



11.4.7.3 Link Control 3 Register (0x1D4)

Bit Location	Attribute	Default Value	Description
0	RsvdP	0b	Perform Equalization. When this bit is 1b and a 1b is written to the Link Retrain register with Target Link Speed set to 8.0 GT/s, the Downstream Port must perform Transmitter Equalization. This bit is RW for Upstream Ports when Crosslink Supported is 1b. This bit is not applicable and is RsvdP for Upstream Ports when the Crosslink Supported bit is 0b. The default value is 0b.
1	RsvdP	0b	Link Equalization Request Interrupt Enable. When Set, this bit enables the generation of interrupt to indicate that the Link Equalization Request bit has been set. This bit is RW for Upstream Ports when Crosslink Supported is 1b. This bit is not applicable and is RsvdP for Upstream Ports when the Crosslink Supported bit is 0b. The default value for this bit is 0b.
31:2	RsvdP	0x00	Reserved.

11.4.7.4 Lane Error Status Register (0x1D8)

The Lane Error Status register consists of a 32-bit vector, where each bit indicates if the corresponding PCIe lane detected an error.

Bit Location	Attribute	Default Value	Description
7:0	RW1CS	0x00	Lane Error Status Bits. 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. The default value of this field is 0b. This field is intended for debug purposes only.
31:8	RsvdZ	0x00	Reserved.

11.4.7.5 Lane Equalization Control Register (0x1DC: 0x1E8)

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.

15

0

Lane (0) Equalization Control Register
Lane (1) Equalization Control Register
...
Lane (Maximum Link Width - 1) Equalization Control Register

Lane [(Maximum Link Width - 1):0] Equalization Control register:



Bit Location	Attribute	Default Value	Description
3:0	RsvdP	0000b	Downstream Port Transmitter Preset. For an Upstream Port if Crosslink Supported is 0b, this field is RsvdP.
6:4	RsvdP	000b	Downstream Port Receiver Preset Hint. For an Upstream Port if Crosslink Supported is 0b, this field is RsvdP.
7	Rsvd	0b	Reserved.
11:8	RO	1111b	Upstream Port Transmitter Preset – Field contains the Transmit Preset value sent or received during Link Equalization. Since crosslink is not supported, the field is intended for debug and diagnostics. It contains the value captured from the associated Lane during Link Equalization. Field is RO. The default value is 1111b.
14:12	HwInit/RO	111b	Upstream Port Receiver Preset Hint – Field contains the Receiver Preset Hint value sent or received during Link Equalization. Field usage varies as follows: Since crosslink is not supported, the field is intended for debug and diagnostics. It contains the value captured from the associated Lane during Link Equalization. Field is RO. The default value is 111b.
15	Rsvd	0b	Reserved.

11.5 Virtual Functions (VFs)

11.5.1 Overview

11.5.1.1 VF to PF Allocation

The X710-TM4 supports up to 128 VFs. The VFs can be allocated to LAN ports in granularity of 16 VFs. Under this constraint, These VFs can be distributed arbitrarily among the different PFs. The distribution is done by NVM settings as the TotalVFs parameter should be stable at enumeration time.

For each of the potential PFs, the following parameters are defined in the NVM:

- PCIe* Configuration Space Control 1.SR-IOV enable - should SR-IOV be exposed for this function.
- Per PF First VF and Last VF - what is the first VF and the last VF allocated to the each function (out of 128). Can be any number between 0 and 128 - 1.

From these parameters the following parameters are derived in the SR-IOV capability structure (see [Section 11.4.4](#) for details):

- The SR-IOV structure is part of the configuration space of a PF only if the GLPCI_CAPSUP.IOV_EN bit is set in the NVM and PF_VT_PFALLOC.VALID field is set for this function.
- InitialVFs = TotalVFs = PF_VT_PFALLOC.LASTVF[n] - PF_VT_PFALLOC.FIRSTVF [n] + 1
- First VF Offset = PF_VT_PFALLOC.FIRSTVF [n]+ 16 - PF# for ARI mode and PF_VT_PFALLOC.FIRSTVF [n]+ 272 - PF# for non ARI mode.

Note: The first VF offset formula is defined so that the RID of a VF is fixed no matter which PF it belongs to.

- VF stride = 1



The First VF and last VF allocated to a PF can be read from the *PF_VT_PFALLOC* registers.
 The total number of enabled virtual functions is reflected in the *GLGEN_PCIFCNCNT* register.

11.5.1.2 Bus-Device-Function Layout

The requester ID allocation of the VF is done using the *First VF Offset* field and the *VF stride* in the IOV structure and is used to do the enumeration of the VFs.

11.5.1.2.1 ARI Mode

The ARI capability allows interpretation of the Device part of the Requester ID as part of the Function part. Thus a single bus can span up to 256 functions.

The allocation of VFs to PF is flexible, there is no relationship between the PF RID and the associated VFs RID.

Table 11-19. RID Per VF - ARI Mode

VF#/PF#	B,D,F	Binary	Notes
PF 0	B,0,0	B,00000,000	PF #0
PF 1	B,0,1	B,00000,001	PF #1
PF 2	B,0,2	B,00000,010	PF #2
...	
PF 15	B,1,7	B,00001,111	PF #15
VF 0	B,2,0	B,00010,000	
VF 1	B,2,1	B,00010,001	
VF 2	B,2,2	B,00010,010	
...	
VF 127	B,17,7	B,10001,111	Last

11.5.1.2.2 Non ARI Mode

When the hierarchy is not ARI capable (VF ARI in SR-IOV Control register is cleared), a non-zero PCI Device Number in the first bus can not be used, thus a second bus is needed to provide enough requester IDs. In this mode, we support only up to eight PFs and the RID layout is as follows:

Table 11-20. RID Per VF - Non ARI Mode

VF#/PF#	B,D,F	Binary	Notes
PF 0	B,0,0	B,00000,000	PF #0
PF 1	B,0,1	B,00000,001	PF #1
PF 2	B,0,2	B,00000,010	PF #2
...	
PF 7	B,0,7	B,00000,111	PF #7



Table 11-20. RID Per VF - Non ARI Mode

VF#/PF#	B,D,F	Binary	Notes
VF 0	B+1,2,0	B+1,00010,000	
VF 1	B+1,2,1	B+1,00010,001	
VF 2	B+1,2,2	B+1,00010,010	
...	
VF 127	B+1,17,7	B+1,10001,111	Last

11.5.1.3 Configuration Space Overview

The configuration space reflected to each of the VF is a sparse version of the physical function configuration space. The following table describes the behavior of each register in the VF configuration space.

Table 11-21. VF PCIe Configuration Space

Section	Offset	Name	VF behavior	Notes
PCI Mandatory Registers	0	Vendor ID	RO – 0xFFFF	
	2	Device ID	RO – 0xFFFF	
	4	Command	Per VF	See Section 11.5.2.3 .
	6	Status	Per VF	See Section 11.5.2.4 .
	8	RevisionID	RO as PF	
	9	Class Code	RO as PF	
	C	Cache Line Size	RO – 0x0	
	D	Latency Timer	RO – 0x0	
	E	Header Type	RO – 0x0	
	F	Reserved	RO – 0x0	
	10 – 27	BARs	RO – 0x0	Emulated by VMM.
	28	CardBus CIS	RO – 0x0	Not used.
	2C	Sub Vendor ID	RO as PF	
	2E	Sub System	RO.Same value for all VFs of each PF	See Section 11.5.2.5 .
	30	Expansion ROM	RO – 0x0	Emulated by VMM.
	34	Cap Pointer	RO – 0x70	Next = MSI-X capability.
	3C	Int Line	RO – 0x0	
	3D	Int Pin	RO – 0x0	
	3E	Max Lat/Min Gnt	RO – 0x0	
MSI-X Capability	70	MSI-X Header	RO – 0xA011	Next = PCIe capability.
	72	MSI-x Message Control	per VF	See Section 11.5.3.1.1 .
	74	MSI-X table Address	RO	See Section 11.5.3.1.2
	78	MSI-X PBA Address	RO	See Section 11.5.3.1.3



Table 11-21. VF PCIe Configuration Space

Section	Offset	Name	VF behavior	Notes
PCIe Capability	A0	PCIe Header	RO – 0x0010	Next = Last capability.
	A2	PCIe Capabilities	RO – as PF	
	A4	PCIe Dev Cap	RO – as PF	
	A8	PCIe Dev Ctrl	RW	As PF apart from FLR – See Section 11.5.3.2.1.
	AA	PCIe Dev Status	per VF	See Section 11.5.3.2.2.
	AC	PCIe Link Cap	RO – as PF	
	B0	PCIe Link Ctrl	RO – 0x0	
	B2	PCIe Link Status	RO – 0x0	
	C4	PCIe Dev Cap 2	RO – as PF	
	C8	PCIe Dev Ctrl 2	RO – 0x0	
	D0	PCIe Link Ctrl 2	RO – 0x0	
	D2	PCIe Link Status 2	RO – 0x0	
	AER Capability	100	AER – Header	RO – 0x15010002
104		AER – Uncorr Status	per VF	See Section 11.5.3.3.1..
108		AER – Uncorr Mask	RO – 0x0	
10C		AER – Uncorr Severity	RO – 0x0	
110		AER – Corr Status	Per VF	See Section 11.5.3.3.2..
114		AER – Corr Mask	RO – 0x0	
118		AER – Cap/Ctrl	RO	See Section 11.5.3.3.3.
11C – 128		AER – Error Log	Shared two logs for all VFs	Same structure as in PF. In case of overflow, the header log is filled with ones.
ARI Capability	150	ARI – Header	0x1A01000E	Next = TPH structure.
	154	ARI – Cap/Ctrl	RO – 0X0	
TPH Requester Capability	0x1A0	TPH - Header	0x1D010017	Next = ACS structure.
	0x1A4	TPH - Capability	RO - 0x00000005	No table reported.
	0x1A8	TPH - Control	per VF	Same structure as in PF.
ACS Capability	0x1D0	ACS - Header	RO - 0x0001000D	Next = Last extended capability.
	0x1D4	ACS - Capability	RO - 0x00000000	

11.5.2 Mandatory Configuration Space

The IOV specification defines the configuration space of the VFs as a mirror of the PF configuration space with the exception of some fields which are implemented per VF.

This section describes the expected handling of the different part of the configuration space for VFs. It deals only with the parts relevant to the X710-TM4 and describes only changes which are not a trivial implementation of the specification.



11.5.2.1 Legacy PCI Configuration Space

The legacy configuration space is allocated to the PF only and emulated for the VFs. A separate set of BARs and one Bus master enable bit is allocated to the whole set of VFs.

All the legacy error reporting bits are emulated for the VF.

11.5.2.2 Memory BARs Assignment

The IOV spec 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.

The only BARs that are useful for the VFs are BAR0 and BAR3, thus only those are replicated.

The following table lists the BARs and the stride used for the VFs:

Table 11-22. VF BARs in X710-TM4

BAR	Type	Usage	Requested Size Per VF
0	Mem	CSR space	
1	Mem	High word of CSR space address	N/A
2	N/A	Not used	N/A
3	Mem	MSI-X	maximum (16K, page size).
4	Mem	High word of MSI-X space address	N/A
5	N/A	Not used	N/A

11.5.2.3 VF Command Register (0x4; RW)

Bit(s)	Initial Value	Rd/Wr	Description
0	0b	RO	IOAE: I/O Access Enable. RO as zero field.
1	0b	RO	MAE: Memory Access Enable. RO as zero field.
2	0b	RW	BME: Bus Master Enable. Disabling this bit prevents the associated VF from issuing any memory or I/O requests. Note that as MSI/MSI-X interrupt messages are in-band memory writes, disabling the bus master enable bit disables MSI/MSI-X interrupt messages as well. Requests other than memory or I/O requests are not controlled by this bit. Note: The state of active transactions is not specified when this bit is disabled after being enabled. The device can choose how it behaves when this condition occurs. Software cannot count on the device retaining state and resuming without loss of data when the bit is re-enabled. Transactions for a VF that has its <i>Bus Master Enable</i> set must not be blocked by transactions for VFs that have their <i>Bus Master Enable</i> cleared.
3	0b	RO	SCM: Special Cycle Enable. Hardwired to 0b
4	0b	RO	MWIE: MWI Enable. Hardwired to 0b.



Bit(s)	Initial Value	Rd/Wr	Description
5	0b	RO	PSE: Palette Snoop Enable. Hardwired to 0b.
6	0b	RO	PER: Parity Error Response. Zero for VFs.
7	0b	RO	WCE: Wait Cycle Enable. Hard wired to 0b.
8	0b	RO	SERRE: SERR# Enable. Zero for VFs.
9	0b	RO	FB2BE: Fast Back-to-Back Enable. Hardwired to 0b.
10	0b	RO	INTD: Interrupt Disable. Hardwired to 0b.
15:11	0b	RO	RSV: Reserved.

11.5.2.4 VF Status Register (0x6; RW)

Bits	Initial Value	Rd/Wr	Description
2:0	0x0	RO	RSV: Reserved.
3	0b	RO	IS: Interrupt Status. Hardwired to 0b.
4	1b	RO	NC: New Capabilities. Indicates that X710-TM4 VFs implement extended capabilities. The X710-TM4 VFs implement a capabilities list, to indicate that it supports MSI-X and PCIe extensions.
5	0b	RO	66E: 66 MHz Capable. Hardwired to 0b.
6	0b	RO	RSV: Reserved
7	0b	RO	FB2BC: Fast Back-to-Back Capable. Hardwired to 0b.
8	0b	RW1C	MPERR: Data Parity Reported.
10:9	00b	RO	DEVSEL: DEVSEL Timing. Hardwired to 0b.
11	0b	RW1C	STA: Signaled Target Abort.
12	0b	RW1C	RTA: Received Target Abort.
13	0b	RW1C	RMA: Received Master Abort.
14	0b	RW1C	SSERR: Signaled System Error.
15	0b	RW1C	DSERR: Detected Parity Error.

11.5.2.5 VF Subsystem ID (0x2E; RO)

This value is loaded from NVM if the `GLPCI_CAPSUP.LOAD_SUBSYS_ID` bit is set. Each VF is loaded from the respective PF's NVM `PFPCI_SUBSYSID.VF_SUB_ID` field (such as all VFs of a specific PF share the same value).

11.5.3 PCI and PCIe Capabilities

The following capability structures are partially replicated in VFs configuration space:

- PCIe capability structure.
- MSI-X capability structure



The following extended capability structures are partially replicated in VFs config space:

Table 11-23. Extended Capabilities List

Address range	Item	Cases Where Capability Does Not Exist	Next Pointer
0x100 - 0x128	Advanced Error Reporting (AER)	None (always present)	Any of the below / 0x000
0x150 - 0x154	Alternative RID Interpretation (ARI)	ARI Enabled bit in NVM is set to 0b	Any of the below / 0x000
0x1A0 - 0x1A8	TPH Requester	TPH Enabled bit in NVM is set to 0b	Any of the below / 0x000
0x1B0 - 0x1B4	Access Control Services (ACS)	ACS Enabled bit in NVM is set to 0b	0x000

11.5.3.1 MSI-X Capability

The MSI-X BAR size is max(16K, page size).

The location and size of the MSI-X vector table and the MSI-X Pending Bits table are determined as follows:

- MSI-X vector table:
 - The MSI-X table structure ([Section 11.3.3.2](#)) 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.
 - Starts at offset 0x0000 from start of BAR.
 - Contains the MSI-X vectors for the VF. The number of entries in the table (N). The maximum value of N is 17 per VF (for the case of up to 32 VFs).
 - The vectors start with the *Vector 0* bit (one per VF), followed by the other vectors allocated to the VF
- MSI-X pending bits table:
 - The PBA structure [[Section 11.3.3.2.2](#)] 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.
 - Starts at half the BAR size (default is offset 0x2000 - 8KB from start of BAR).
 - Contains the pending bits for the VF. The VF is allocated one 64-bit register for a maximum of 17 bits.
 - The bits start with the *Vector 0* bit (one per VF), followed by bits for the other vectors allocated to the VF.

11.5.3.1.1 VF MSI-X Control Register (0x72; RW)



Bits	Initial Value	Rd/Wr	Description
10:0	0x004 (5 vectors)	RO	TS: Table Size (N-1). N varies with the number of virtual functions. This field is loaded from NVM. It is reflected in the GLPCI_CNF2.MSI_X_VF_N CSR field.
13:11	0x0	RO	RSV: Reserved.
14	0b	RW	Mask: Function Mask.
15	0b	RW	En: MSI-X Enable.

11.5.3.1.2 MSI-X Address Register (0x74; RO)

Bits	Default	Type	Description
2:0	0x3	RO	Table BIR. Indicates which one of a function’s BARs, beginning at 0x10 in the configuration space, is used to map the function’s MSI-X table into the memory space. while BIR values: 0...5 correspond to BARs 0x10...0x 24 respectively.
31:3	0x000	RO	Table offset. Used as an offset from the address contained by one of the function’s BARs to point to the base of the MSI-X vectors address. The lower three BIR bits are masked off (set to zero) by software to form a 32-bit Qword-aligned offset.

11.5.3.1.3 MSI-X PBA Register (0x78; RO)

Bits	Default	Type	Description
2:0	0x3	RO	PBA BIR. Indicates which one of a function’s BARs, located beginning at 0x10 in configuration space, is used to map the function’s MSI-X PBA into memory space. A BIR value of three indicates that the PBA is mapped in BAR 3.
31:3	0x400	RO	PBA Offset. Used as an offset from the address contained by one of the function’s BARs to point to the base of the MSI-X PBA. The lower three PBA BIR bits are masked off (set to zero) by software to form a 32-bit Qword-aligned offset. This value is changed by hardware to be half of the requested BAR size.

11.5.3.2 PCIe Capability Registers

The device control and device status registers have some fields which are specific per VF.

11.5.3.2.1 VF Device Control Register (0xA8; RW)

Bits	Rd/Wr	Default	Description
0	RO	0b	Correctable Error Reporting Enable. Zero for VFs.
1	RO	0b	Non-Fatal Error Reporting Enable. Zero for VFs.



Bits	Rd/Wr	Default	Description
2	RO	0b	Fatal Error Reporting Enable. Zero for VFs.
3	RO	0b	Unsupported Request Reporting Enable. Zero for VFs.
4	RO	0b	Enable Relaxed Ordering. Zero for VFs.
7:5	RO	0b	Max Payload Size. Zero for VFs.
8	RO	0b	Extended Tag field Enable.
9	RO	0b	Reserved.
10	RO	0b	Aux Power PM Enable. Zero for VFs.
11	RO	0b	Enable No Snoop. Zero for VFs.
14:12	RO	000b	Max Read Request Size. Zero for VFs.
15	RW	0b	Initiate Function Level Reset. Specific to each VF.

11.5.3.2.2 VF Device Status Register (0xAA; RO)

Bits	Rd/Wr	Default	Description
0	R/W1C	0b	Correctable Detected. Indicates status of correctable error detection.
1	R/W1C	0b	Non-Fatal Error Detected. Indicates status of non-fatal error detection.
2	R/W1C	0b	Fatal Error Detected. Indicates status of fatal error detection.
3	R/W1C	0b	Unsupported Request Detected. Indicates that the X710-TM4 received an unsupported request. This field is separate per VF. However, in case where an error cannot be associated with a VF, this bit is set in all PFs and VFs.
4	RO	0b	Aux Power Detected. Zero for VFs.
5	RO	0b	Transactions Pending. Specific per VF. When set, indicates that a particular function (PF or VF) has issued non-posted requests that have not been completed. A function reports this bit cleared only when all completions for any outstanding non-posted requests have been received.
15:6	RO	0x00	Reserved.

11.5.3.3 AER Registers

The following registers in the AER capability have a different behavior in a VF function.

Note that unlike the PF AER registers, these registers are not sticky since the VF is reset on FLR and on in-band reset.

11.5.3.3.1 Uncorrectable Error Status Register (0x104; RW1C)

Bit Location	Attribute	Default Value	Description
3:0	RO	0x0	Reserved.
4	RO	0b	Data Link Protocol Error Status. Hardwired to 0b.
5	RO	0b	Surprise Down Error Status. Hardwired to 0b.



Bit Location	Attribute	Default Value	Description
11:6	RO	0x0	Reserved.
12	RW1C	0b	Poisoned TLP Status.
13	RO	0b	Flow Control Protocol Error Status. Hardwired to 0b.
14	RW1C	0b	Completion Timeout Status.
15	RW1C	0b	Completer Abort Status.
16	RW1C	0b	Unexpected Completion Status.
17	RO	0b	Receiver Overflow Status. Hardwired to 0b.
18	RO	0b	Malformed TLP Status. Hardwired to 0b.
19	RO	0b	ECRC Error Status. Hardwired to 0b.
20	RW1C	0b	Unsupported Request Error Status. When caused by a function that claims a TLP.
21	RO	0b	ACS Violation Status. Hardwired to 0b.
31:22	RO	0x0	Reserved.

11.5.3.3.2 Correctable Error Status Register (0x110; RW1C)

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.

Bit Location	Attribute	Default Value	Description
0	RO	0b	Receiver Error Status. Hardwired to 0b.
5:1	RO	0x0	Reserved.
6	RO	0b	Bad TLP Status. Hardwired to 0b.
7	RO	0b	Bad DLLP Status. Hardwired to 0b.
8	RO	0b	REPLAY_NUM Rollover Status. Hardwired to 0b.
11:9	RO	0x0	Reserved.
12	RO	0b	Replay Timer Timeout Status. Hardwired to 0b.
13	RW1C	0b	Advisory Non-Fatal Error Status.
31:14	RO	0b	Reserved.

11.5.3.3.3 Advanced Error Capabilities and Control Register (0x118; RO)

Bit Location	Attribute	Default Value	Description
4:0	ROS	0b	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.



Bit Location	Attribute	Default Value	Description
5	RO	0b	ECRC Generation Capable. If set, this bit indicates that the function is capable of generating ECRC. This bit is loaded from the NVM. It is reflected in the GLPCI_CAPSUP register.
6	RO	0b	ECRC Generation Enable. When set, ECRC generation is enabled. Hardwired to 0b. The PF setting applies to the VF.
7	RO	0b	ECRC Check Capable. If set, this bit indicates that the function is capable of checking ECRC. This bit is loaded from the NVM. It is reflected in the GLPCI_CAPSUP register.
8	RO	0b	ECRC Check Enable. When set, ECRC checking is enabled. Hardwired to 0b. The PF setting applies to the VF.
9	RO	0b	Multiple Header Recording Capable. Not supported. Hardwired to 0b.
10	RO	0b	Multiple Header Recording Enable. Not supported. Hardwired to 0b.
11	RsvdP	0b	TLP Prefix Log Present. Not supported. Hardwired to 0b.
15:12	RO	0x0	Reserved.



12.0 Reliability, Diagnostics and Testability

12.1 Reliability

12.1.1 ECC Support and ECC Error Flow

Memories in the X710-TM4 are protected by ECC (ECC bits are added to the memory on write and compare on read). There are two types of ECC errors:

- Correctable ECC error — When the memory line has a single bit error, the ECC mechanism corrects it. This is done within the memory shell/wrapper and the flow continues as normal.
- Uncorrectable ECC error — When the memory line read has more than a single ECC error, it cannot be recovered by the ECC mechanism. The text that follows describes how the X710-TM4 reacts to such an event.

Reporting of ECC errors is as follows:

- The GL_CRITERMODMASK2 register masks the reporting of ECC errors per block.
 - If enabled, a correctable error is indicated by the ITR_CAUSE_MEM_0_STATUS.ECC_FIX bit and an uncorrectable error is indicated by the ITR_CAUSE_MEM_0_STATUS.ECC_ERR bit.
- If the ECC_ENA.ECC_ENA is set to 1b, an uncorrectable ECC error sets the ECC_ERR interrupt cause (for all PFs).
- The *_ECC_COR_ERR and *_ECC_UNCOR_ERR per-block registers count the occurrence of correctable and uncorrectable errors, respectively (the * symbol stands for the block name).

Uncorrectable errors are handled as follows:

- Device blocks data from going to an external link and blocks any new PCIe master transaction, but the erroneous transaction may reach host memory.
- Recovery then depends on the memory where the error happens:
 - An error takes place in the core or global domains
 - If the GLGEN_RSTCTL.ECC_RST_EN bit is set to 1b, the X710-TM4 generates a GLOBR reset
 - An error takes place in the EMP
 - If the GLMNG_WD_ENA.ECC_RST_ENA bit is set to 1b, the EMP generates an EMPR reset
 - An error takes place in FLEEP (Shadow RAM)
 - An ECC error check in the shadow RAM is enabled via the *Shadow RAM ECC Enable* bit in the NVM
 - The FLEEP reloads the shadow RAM from the NVM.
 - If the GLGEN_RSTCTL.ECC_RST_EN bit is set to 1b, the X710-TM4 generates a GLOBR reset (which in turn reloads the relevant sections into hardware)
 - An error takes place in the PCIe domain
 - If an error is in data to be sent out, the TLP is sent with an EDB
 - Else, the link goes to a link-down state
 - If the GLMNG_WD_ENA.ECC_RST_ENA bit is set to 1b, the EMP generates an EMPR reset



- Else, if the GLGEN_RSTCTL.ECC_RST_EN bit is set to 1b, the X710-TM4 generates a GLOBR reset

12.2 Link Loopback Operations

Loopback operations are supported by the X710-TM4 to assist with system and device debug. Loopback operation can be used to test transmit and receive aspects of software device drivers, as well as to verify electrical integrity of the connections between the X710-TM4 and the system (such as PCIe bus connections)

12.2.1 MAC Loopback

MAC loopback can be enabled using the Set Loopback Modes command.

12.2.2 BASE-T PCS Loopback Support

- When link is on before the Set Loopback Mode AQ command is sent by the host, the PCS loopback should be established as follows:
 - When the line speed is 10 Gb/s, 5 Gb/s or 2.5 Gb/s the loopback speed should be 10 Gb/s.
 - When the line speed is 1 Gb/s the loopback speed should be 1 Gb/s.
 - When the line speed is 100 Mb/s the loopback speed should be 100 Mb/s.
- When link is off before the Set Loopback Mode AQ command is sent by the host, the PCS loopback should be established according to previously configured maximum advertisement speed:
 - If the maximum available speed, according to the active capabilities is 10 Gb/s, 5 Gb/s or 2.5 Gb/s, the loopback speed should be 10 Gb/s.
 - If the maximum available speed, according to the active capabilities is 1 Gb/s, the loopback speed should be 10 Gb/s.
 - If the maximum available speed, according to the active capabilities is 100 Mb/s, the loopback speed should be 100 Mb/s.
- All other options should be ignored.
- Firmware should return an I40E_AQ_RC_OK (0x0) response when loopback is established according to the previous rules.



12.2.3 Base-T External Loopback Support

In external loopback mode, the device should only support 1 Gb/s line speed.

All other options should be ignored.

Firmware should return an I40E_AQ_RC_OK (0x0) response when 1 Gb/s loopback is established according to the previous rules.

12.3 Get/Clear Log

Retrieving the logger messages via a polling mechanism is also possible, by using the appropriate AQ command.

12.4 NVM Recovery Mode

NVM recovery mode is intended to recover from a misconfiguration, which can be caused by an interrupted firmware update, power failure, host software access or interrupted MC configuration access. This misconfiguration prevents firmware from executing successful initialization flows. These flows enable the software device driver/MC to fix the misconfiguration and cause firmware to initialize the correct data path.

Note: Device firmware implements a recovery mode that requires driver and firmware update tool software assistance to complete the recovery process. For a full discussion of recovery mode, Refer to the Intel® Ethernet Controller X710-TM4/AT2 NVM Recovery Mode Application Note.

During NVM recovery mode, firmware sets the *Recovery Mode* bit in FWSTS. While in the mode, firmware only supports a limited set of Admin commands listed in [Table 12-1](#). These minimal commands enable software tools to recover the NVM.

Table 12-1. Admin Commands Supported (NVM Recovery Mode)

Command	Opcode	Comment
Get Version	0x0001	
Set PF Context	0x0004	In recovery mode, only PF0 is allowed. Any other value returns EINVAL and PF0 is used.
Request Resource Ownership	0x0008	
Request Resource Ownership	0x0009	
Discover Function Capabilities	0x000A	
Discover Device Capabilities	0x000B	
NVM Read	0x0701	
NVM Erase	0x0702	
NVM Update	0x0703	
Write Alternate - Direct	0x0900	
Write Alternate - Indirect	0x0901	



Table 12-1. Admin Commands Supported (NVM Recovery Mode)

Command	Opcode	Comment
Read Alternate - Direct	0x0902	
Read Alternate - Indirect	0x0903	
Done Alternate - Write	0x0904	
Write Alternate - Set OEM Mode	0x0905	
Clear Port Alternate	0x0906	
Debug Dump Internal Data	0xFF08	
Debug Read Register	0xFF03	

Note: Firmware exits NVM recovery mode only after the NVM update completes by NVM update software.

12.4.1 Detect NVM Recovery Mode

To detect NVM recovery mode, driver software:

1. Reads FWSTS or reads the *Operation Mode* bit via the Discover Function/Device Capabilities Admin Queue Command (AQC).
2. Notifies users an NVM recovery is needed.
3. Gracefully handles the error flow to avoid an operating system crash.

12.4.2 NVM Update Tool (NUT)/Recovery Tool

This tool performs NVM recovery using the following method:

1. Accesses the device using PF0.
2. Detects NVM recovery mode.
3. Extracts factory ETID, PBA and MAC addresses from SW-Data-Recovery in the NVM.
4. Restores NVM update image to:
 - a. The latest factory image based on ETID and MAC addresses (default).
 - b. Last known good ETID, assuming having this lookup information and no security-revision increment was ever completed (user option).



13.0 Electrical/Mechanical Specification

13.1 Introduction

This section describes the X710-TM4 electrical and mechanical characteristics, including:

- Operating conditions
- Power delivery
- Power dissipation
- DC/AC specifications
- Package
- Supported devices

13.2 Operating Conditions

13.2.1 Absolute Maximum Ratings

Table 13-1. Absolute Maximum Ratings

Symbol	Parameter	Min	Max	Units
T_{case}	Case Temperature Under Bias	0	107	°C
$T_{storage}$	Storage Temperature Range	-50	150	°C
V_i	3.3V I/O input Voltage	VSS-0.5	4	V
VCC3P3	3.3V Digital/Analog Periphery Supply Voltage	VSS-0.5	4	V
VCCD	Core/Periphery/Analog Supply Voltage	VSS-0.2	1.19	V
ICC3P3	3.3V Periphery Supply Current	-	1.1	A
ICCD	VCCD/VCCDA Core/Periphery/Analog Supply Current	-	7.5	A

Note: Stresses above those listed in the table can cause permanent device damage. These values should not be used as limits for normal device operation. Exposure to absolute maximum rating conditions for an extended period of time can affect device reliability.



13.2.2 Recommended Operating Conditions

Table 13-2. Recommended Operating Conditions

Symbol	Parameter	Min	Typ	Max	Units
Ta	Operating Temperature Range Commercial (Ambient) ¹	0		55	°C
Tj	Junction Temperature			110	°C
VCC3P3	3.3V Digital/Analog Power Supply	3.14	3.3	3.46	V
VCCD	Digital/Analog Power Supply	0.87	0.92	0.96	V
AVDDH/VDDIO ²	PHY Analog High and I/O Voltages (tolerance 3%)	1.8236	1.88	1.9364	V
AVDDL ³	PHY Analog Low (tolerance 5%)	0.95	1	1.05	V
DVDD ⁴	PHY Digital Core (tolerance 5%)	0.95	1	1.05	V

1. See [Section 14.0](#).
2. All analog high voltage at 1.88V and VDDIO at 1.88V.
3. All analog low voltage and XFI at 1.0V.
4. Digital voltage at 1.0V.

Note:

- For normal device operation, adhere to the limits listed in the previous table. Sustained operation of a device at conditions exceeding these values, even if they are within the absolute maximum rating limits, can result in permanent device damage or impaired device reliability. Device functionality to stated DC and AC limits is not guaranteed if conditions exceed recommended operating conditions.

External Heat Sink (EHS) is needed.

Refer to [Section 14.0](#) for a description of the allowable thermal environment.

13.3 Power Delivery

13.3.1 Power Supply Specification

Table 13-3. External 3.3V Power Supply Specification

VCC3P3/VCC3P3_A (3.3V) Parameters					
Title	Description	Min	Max	Units	
Rise Time	Time from 10% to 90% mark	0.1	100	ms	
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV	
Slope	Ramp rate at any given time between 10% and 90% Min: $0.8 \cdot V(\text{min}) / \text{rise time}(\text{max})$ Max: $0.8 \cdot V(\text{max}) / \text{rise time}(\text{min})$	24	28,800	V/S	

**Table 13-3. External 3.3V Power Supply Specification (Continued)**

VCC3P3/VCC3P3_A (3.3V) Parameters				
Operational Range	Voltage range for normal operating conditions	3.3-5%	3.3+5%	V
Ripple	Maximum voltage ripple (peak to peak)	N/A	70	mV
Overshoot	Maximum overshoot allowed	N/A	100	mV
Overshoot Settling Time	Maximum overshoot allowed duration. (At that time delta voltage should be lower than 5 mV from steady state voltage)	N/A	0.05	ms

Table 13-4. External VCCD Power Supply Specification

VCCD/VCCD_A Parameters				
Title	Description	Min	Max	Units
Rise Time	Time from 10% to 90% mark	0.1	10	ms
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: $0.8 \cdot V(\text{min}) / \text{rise time}(\text{max})$ Max: $0.8 \cdot V(\text{max}) / \text{rise time}(\text{min})$	N/A	7120	V/S
Operational Range	Voltage range for normal operating conditions	0.87	0.96	$\pm 5\% V$
Ripple	Maximum voltage ripple (peak to peak)	N/A	20	mV
Overshoot	Maximum overshoot allowed	N/A	50	mV
Overshoot Duration	Maximum overshoot allowed duration. (At that time delta voltage should be lower than 5 mV from steady state voltage)	0	0.05	ms

Note: Inaccuracy of the power supply should be up to 2%.

13.3.1.1 Power On/Off Sequence

The following relationships between the rise time of the different power supplies should be maintained at all times when external power supplies are in use to avoid risk of either latch-up or forward-biased internal diodes:

- At power-on and after 3.3V reaches 90% of its final value, the VCCD voltage rail is allowed 100 ms to reach its final operating voltage. Once the VCCD power supply reaches 80% of its final value the 3.3V power supply should always be above 80% of its final value until power down.
- After 3.3V and VCCD rails reaches 65%-90% of its final value, the VCCA voltage automatically rises. The final value of VCCA is reached only after a power on internal calibration period.
- For power down, it is recommended to turn off all rails at the same time and allow voltage to decay.
- For port 0 and port 1 1.0V and 1.88V rails, Intel recommends that they are brought up at the same time, or that 1.88V rails are brought up before the 1.0V rails.
- The ICM center tap voltage and AVDDH should have the same power source.

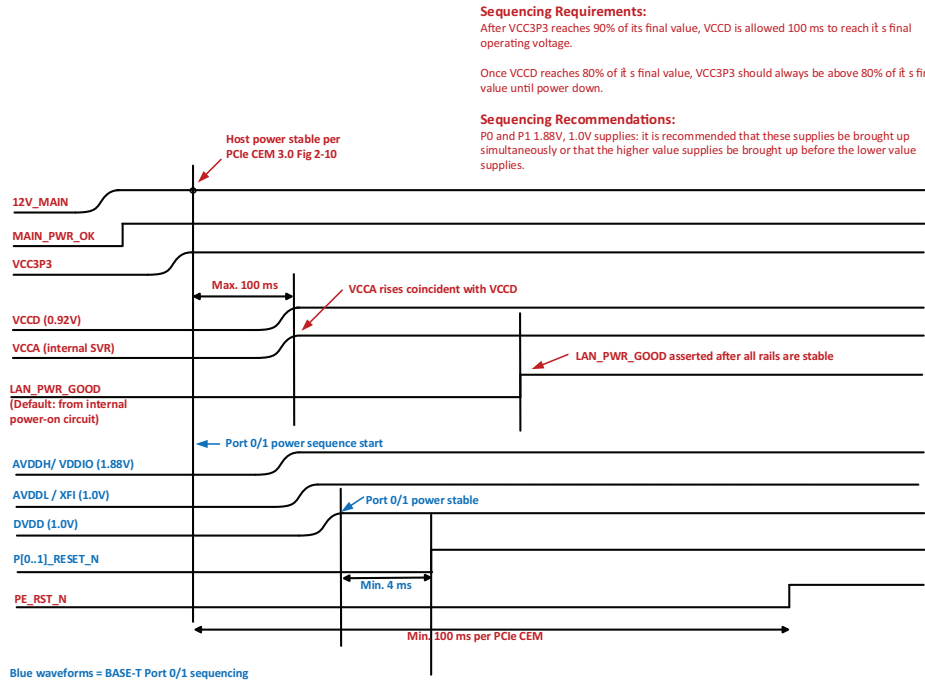


Figure 13-1. Power On/Off Sequence Diagram

13.3.2 In-rush Current

Note: The values provided are a reference. The In-rush current is largely a function of the raw capacitance of the power plane and the dV/dt (slew rate) of the input to that plane. Caution should be taken in the design of the power supply to accommodate these factors for the specific design application.

Table 13-6. In-rush Current¹

Voltage Rail	In-rush Current
VCC3P3 [A]	0.5
VCCD [A]	5.7
AVDDH [A]	0.8
DVDD [A]	2.9
AVDDL [A]	0.85

1. Maximum conditions: T_J = 110 °C during power up link training to establish 10 Gb/s link speed.



13.4 Power Dissipation

The following tables list the targets for device power. The numbers listed apply to device current and power and do not include power losses on external components.

Power numbers are provided in two modes:

Maximum power — Power of the device at worst-case operation conditions. Power is measured on fast silicon, normal voltage and $T_j=110\text{ °C}$ ($T_j\text{-Max}$). This power should be used for thermal and power supply design.

Typical power — Power of the device at normal operation conditions. Power is measured on typical silicon, normal voltage and $T_j=80\text{ °C}$.

13.4.1 Maximum Power (TDP)

Table 13-7. Maximum Power

Link Speed	4x10 GbE	2x10GBASE-T
VCC3P3 [A]	0.617	0.561
VCCD [A]	3.717	3.544
AVDDH [A]	0.781	0.781
DVDD [A]	2.014	2.058
AVVDL [A]	0.749	0.754
Power [W]	9.748	9.462

Note: Maximum conditions: fast material, $T_j = 110\text{ °C}$ and continuous network traffic at link speed.

13.4.2 Typical Power

Table 13-8. Typical Active Power

Link Speed	4x10 GbE	2x10 GBASE-T
VCC3P3 [A]	0.529	0.487
VCCD [A]	2.561	2.213
AVDDH [A]	0.740	0.741
DVDD [A]	2.211	2.179
AVDDL [A]	0.639	0.634
Power [W]	8.342	7.850

Note: Typical conditions: Typical material, $T_j = 80\text{ °C}$, normal voltages and continuous network traffic at link speed.



Table 13-9. Typical Idle Power Without EEE

Link Speed	4x10 GbE	2x10 GBASE-T
VCC3P3 [A]	0.498	0.478
VCCD [A]	2.107	1.955
AVDDH [A]	0.477	0.477
DVDD [A]	1.969	1.937
AVDDL [A]	0.628	0.626
Power [W]	7.079	6.837

Note: Typical conditions: Typical material, T_J = 80 °C, normal voltages and no traffic without EEE.

Table 13-10. Typical Idle Power With EEE

Link Speed	4x10 GbE	2x10 GBASE-T
VCC3P3 [A]	0.526	0.475
VCCD [A]	2.067	1.956
AVDDH [A]	0.477	0.477
DVDD [A]	0.702	0.709
AVDDL [A]	0.604	0.604
Power [W]	5.840	5.576

Note: Typical conditions: Typical material, T_J = 80 °C, normal voltages and no traffic with EEE.

Table 13-11. Typical D3 Cold With Wake-up Enable

Link Speed	4x1 GbE	2x1 GbE	4x100 Mb/s	2x100 Mb/s
VCC3P3 [A]	0.410	0.388	0.395	0.354
VCCD [A]	1.390	1.157	1.317	1.149
AVDDH [A]	0.578	0.577	0.254	0.254
DVDD [A]	0.392	0.394	0.328	0.324
AVDDL [A]	0.403	0.403	0.399	0.396
Power [W]	4.514	4.228	3.718	3.423

Note: Typical conditions: T_J = 80 °C, normal voltages, all ports link up with no traffic and PE_RST_N asserted.



Table 13-12. D3 Cold With No Wake-up (Link Disabled)

Link Speed	No Link
VCC3P3 [A]	0.495
VCCD [A]	1.807
AVDDH [A]	0.210
DVDD [A]	0.217
AVDDL [A]	0.421
Power [W]	4.331

Note: Conditions: fast material, T_J = 80 °C, normal voltages, no link and PE_RST_N asserted (S0).

Table 13-13. D3 Cold With No PCIe and No Wake-up (Link Disabled)

Link Speed	No Link
VCC3P3 [A]	0.393
VCCD [A]	1.337
AVDDH [A]	0.210
DVDD [A]	0.219
AVDDL [A]	0.421
Power [W]	3.561

Note: Conditions: fast material, T_J = 80 °C, normal voltages, no link and no PCIe (S5).

13.4.2.1 BASE-T Power Supply Rail Operating Conditions (Per-port)

Table 13-14. 10GBASE-T

Description	Voltage (V) +3%	Current (A)	Power (W)
Center tap (analog voltage at 1.88V ±3%)	1.936	0.186	0.360
AVDDH (all analog high voltage at 1.88V ±3%)	1.936	0.154	0.298
AVDDL (all analog low voltage and XFI at 0.98V ±3%)	1.01	0.450	0.454
DVDD (digital voltage at 0.98V ±3%)	1.01	1.317	1.330
Total power per port (W)			2.442



Table 13-15. 5GBASE-T

Description	Voltage (V) +3%	Current (A)	Power (W)
Center tap (analog voltage at 1.88V ±3%)	1.936	0.222	0.429
AVDDH (all analog high voltage at 1.88V ±3%)	1.936	0.155	0.299
AVDDL (all analog low voltage and XFI at 0.98V ±3%)	1.01	0.434	0.438
DVDD (digital voltage at 0.98V ±3%)	1.01	0.775	0.782
Total power per port (W)			1.949

Table 13-16. 2.5GBASE-T

Description	Voltage (V) +3%	Current (A)	Power (W)
Center tap (analog voltage at 1.88V ±3%)	1.936	0.222	0.429
AVDDH (all analog high voltage at 1.88V ±3%)	1.936	0.154	0.298
AVDDL (all analog low voltage and XFI at 0.98V ±3%)	1.01	0.425	0.429
DVDD (digital voltage at 0.98V ±3%)	1.01	0.526	0.531
Total power per port (W)			1.687

Table 13-17. 1000BASE-T

Description	Voltage (V) +3%	Current (A)	Power (W)
Center tap (analog voltage at 1.88V ±3%)	1.936	0.209	0.404
AVDDH (all analog high voltage at 1.88V ±3%)	1.936	0.055	0.106
AVDDL (all analog low voltage and XFI at 0.98V ±3%)	1.01	0.281	0.284
DVDD (digital voltage at 0.98V ±3%)	1.01	0.294	0.297
Total power per port (W)			1.091

Table 13-18. 100BASE-TX

Description	Voltage (V) +3%	Current (A)	Power (W)
Center tap (analog voltage at 1.88V ±3%)	1.936	0.055	0.106
AVDDH (all analog high voltage at 1.88V ±3%)	1.936	0.040	0.076
AVDDL (all analog low voltage and XFI at 0.98V ±3%)	1.01	0.259	0.261
DVDD (digital voltage at 0.98V ±3%)	1.01	0.221	0.224
Total power per port (W)			0.668



13.5 SVR Board Connectivity Guidelines

The voltage regulator for VCCA is implemented inside the device, and only the power-FET and filters are needed to be placed externally.

The VCCD and Port0/1 1.0V and 1.88V supplies require external SVRs.

[Section 13.5.1](#) and [Section 13.6](#) describe how the VCCA and VCCD supplies should be connected.

13.5.1 VCCA Connectivity

The VCCA analog rail should be connected through an external inductor to the switching node of the on-die switching voltage regulator: SVR_IND (pin V2 of the device). This switching voltage regulator takes 3.3V as input and operates at a switching frequency of 1 MHz.

13.5.1.1 VCCA SVAR Bill of Material (BOM)

The following are the BOM guidelines.

- Output Bulk Capacitors (on VCCA) — Total capacitance should be 50 μ F. It is recommended to use these four capacitors 10 μ F X7R or two capacitors 22 μ F X7R. The total ESR < 1m Ω and total Z < 1.5m Ω @ F=1MHz.
- Input Bulk Capacitors (on VCC3P3) — Total capacitance should be 40-100 μ F. It is recommended to use a minimum of four capacitors 10 μ F X7R or a minimum of two capacitors 22 μ F X7R. The total ESR << 1m Ω and total Z << 1.5m Ω @ F=1MHz.
- Power Inductor — L= 1 μ H +/- 20%, I_{rated} > 1.6A, I_{sat} > 2.2A, R_{dc} < 15m Ω . The DCR should be low as possible in order to decrease the inductor losses and achieving higher SVR efficiency. The loss caused by this DCR is $P=I^2 R$ and decreases the efficiency by $(I_{out}^2 \times RDC)/P_{in}$ (%). It is recommended to use a Coilcraft* inductor XFL4020-102ME or other equivalent. A shielded inductor is recommended for better EMI performance.
- Decoupling Low ESL Capacitors (Optional) — To avoid high frequency noises on inputs/outputs, there is an option to use low ESL capacitors 10 nF or 100 nF close to the SVR balls. Using them or not depends on the specific board layout and specific system requirements.

13.5.1.2 VCCA Board Connectivity

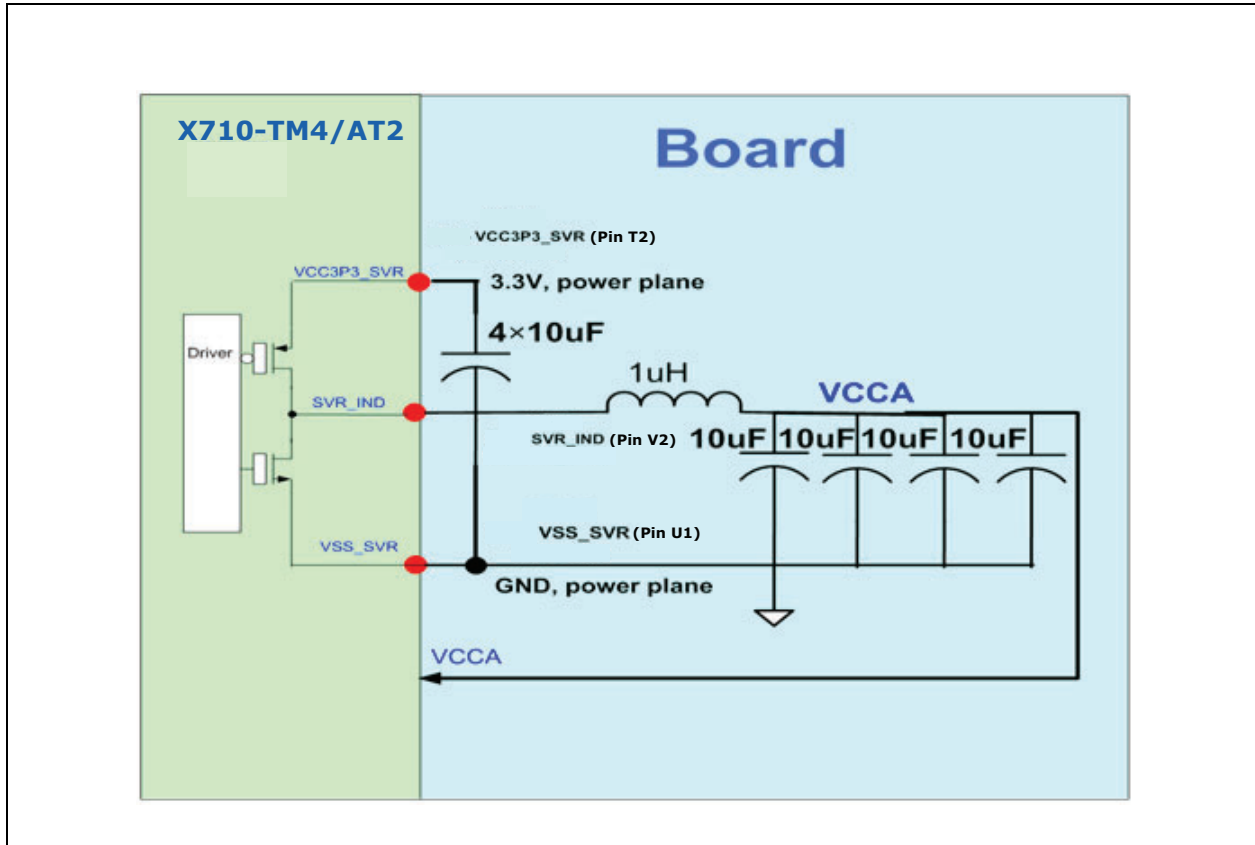


Figure 13-2. X710-TM4 Analog SVR Board Connectivity Schematic

13.5.1.3 VCCA Layout Guideline

All SVR components should be located on the top layer and close as possible to their balls.

The connection between the SVR traces/planes to VCCA, GND and VCC3p3 should be out of the SVR area and far from the balls on the other side (See [Figure 13-2](#)).

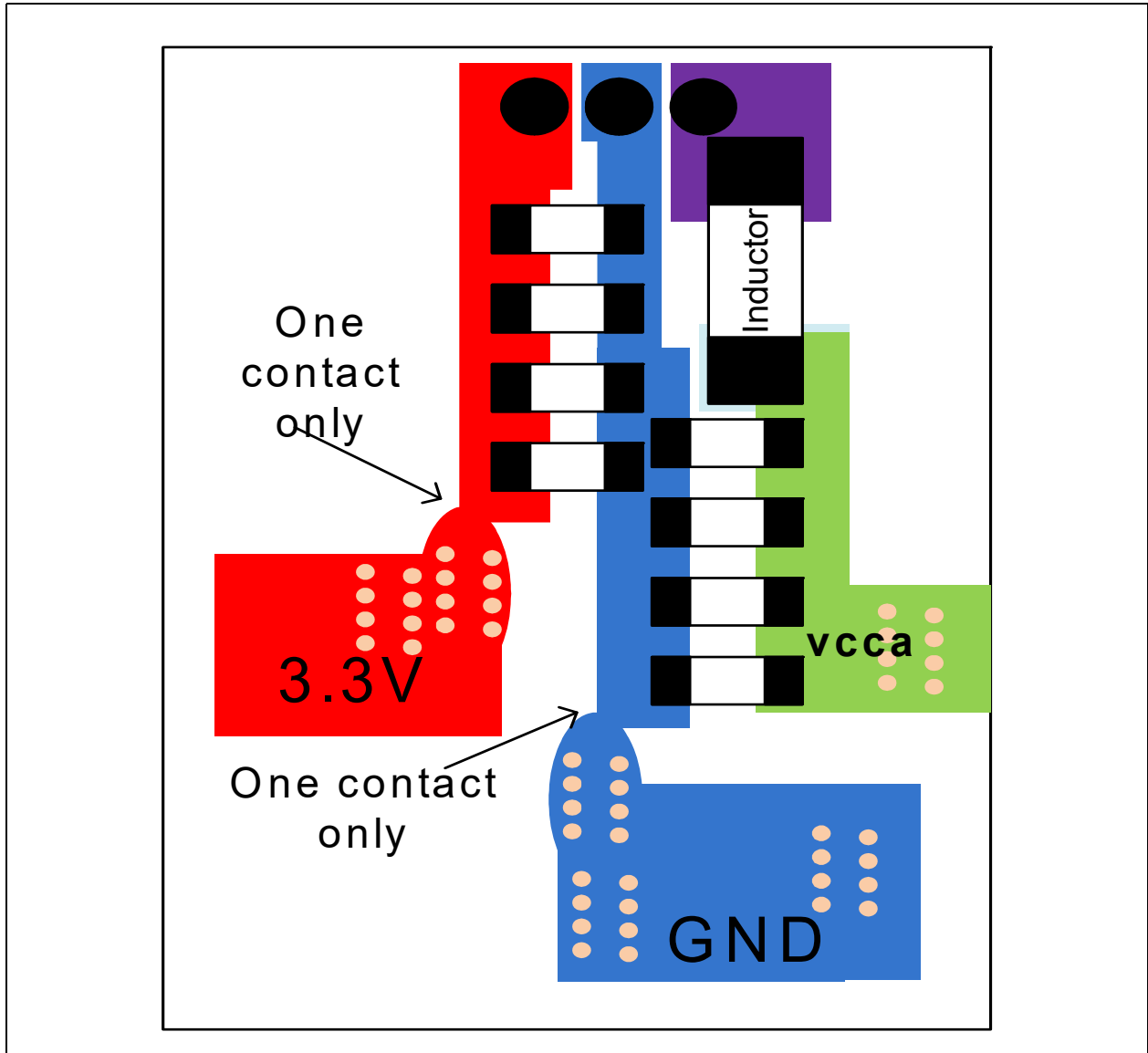


Figure 13-3. Analog SVR Layout Guidelines (VCCA)



13.6 DC/AC Specification

13.6.1 Digital I/O DC Specifications

Table 13-19. Digital Functional 3.3V I/O DC Electrical Characteristics

Symbol	Parameter	Conditions	Min	Max	Units	Note
VOH	Output High Voltage		2.4		V	
VOL	Output Low Voltage			0.4	V	
IOH	Output High Current		12		mA	
IOL	Output High Current		12		mA	
VIH	Input High Voltage		2.0	3.45	V	
VIL	Input Low Voltage		-0.3	0.8	V	
Iil	Input Current ¹	VI=3.3V / 0V		10	μA	
PU	Internal pull-up		54	110	KΩ	
Cin	Pin capacitance		1.5	1.7	pF	[1]

1. Input leakage current.

13.6.1.1 Open Drain I/O DC Specification

This section applies to SMBD, SMBCLK, SMBALRT_N, PE_WAKE_N and MDIO[3:0].

Table 13-20. Open Drain I/O DC Characteristics

Symbol	Parameter	Condition	Min	Max	Units	Note
Vih	Input High Voltage		2.0	3.45	V	
Vil	Input Low Voltage		-0.3	0.8	V	
Ileakage	Output Leakage Current	$0 \leq V_{in} \leq V_{CC3P3 \text{ max}}$		10	μA	1
Vol	Output Low Voltage	@ Ipullup = 4 mA	0.4		V	
Cin	Input Pin Capacitance		2.9	3.3	pF	2
Ioffsm	Input leakage Current	VCC3P3 off or floating		10	μA	1
IOL	Output Current Low		6		mA	
Pull_Up	Total equivalent pull up resistance per OD output		5K		ohm	

1. Device must meet this specification whether powered on or powered off.
2. C_{load} should be calculated according to the external pull-up resistor and the frequency.

The buffer specification meets the SMBus specification requirements defined at: www.smbus.org.



13.6.1.2 NC-SI I/O DC Specification

Table 13-21. NC-SI I/O DC Characteristics

Parameter	Symbol	Conditions	Min.	Typ.	Max	Units
Bus High Reference	Vref ¹		3.0	3.3	3.6	V
Signal Voltage Range	Vabs		-0.3		3.765	V
Input Low Voltage	Vil				0.8	V
Input High Voltage	Vih		2.0			V
Output Low Voltage	Vol	Iol = 4mA, Vref = Vref _{min}	0		0.4	V
Output High Voltage	Voh	Iol = -4mA, Vref = Vref _{min}	2.4		Vref	V
Input High Current	Iih	Vin = 3.6V, Vref = 3.6V	0		200	μA
Input Low Current	Iil	Vin = 0V, Vref = Vref _{min} to Vref _{max}	-20		0	μA
Input High Current	Ioh		12			mA
Input Low Current	Iol		12			mA
Clock Midpoint Reference Level	Vckm				1.4	V
Cin	Pin capacitance		1.5	1.7	pF	[1]
Leakage Current for Output Signals in High-Impedance State	Iz	0 ≤ Vin ≤ Vih _{max} Vref = Vref _{max}	-20		20	μA

1. Vref = Bus high reference level. This parameter replaces the term supply voltage since actual devices might have internal mechanisms that determine the operating reference for the sideband interface that are different from the devices overall power supply inputs. Vref is a reference point that is used for measuring parameters such as overshoot and undershoot and for determining limits on signal levels that are generated by a device. In order to facilitate system implementations, a device must provide a mechanism (such as a power supply pin, internal programmable reference, or reference level pin) to enable Vref to be set to within 20 mV of any point in the specified Vref range. This is to enable a system integrator to establish an inter-operable Vref level for devices on the sideband interface. Although the NC-SI specification defines Vref_{max} up to 3.6V, the X710-TM4 supports Vref_{max} up to 3.46V (3.3V +5%).

13.6.2 Digital I/F AC Specifications

13.6.2.1 Digital I/O AC Specifications

Table 13-22. Digital 3.3V I/O AC Electrical Characteristics

Parameters	Description	Min	Max	Condition	Note
F _{max}	Maximum Operating Frequency		100 MHz 300 MHz	100 MHz at 25 pF load 300 MHz at 8 pF load	
T _{or}	Output Rise Time	0.5 ns	6 ns		
T _{of}	Output Fall Time	0.5 ns	6 ns		



13.6.2.2 SMBus and I²C AC Specifications

The X710-TM4 meets the SMBus AC specification as defined in SMBus specification version 2, section 3.1.1 (<http://www.smbus.org/specs/>) and the I²C specification.

The X710-TM4 also supports a 400 KHz SMBus and I²C (as a slave) and meets the specifications listed in Table 13-23 table:

Table 13-23. Support for 400 KHz SMBus

Symbol	Parameter	Min	Typ	Max	Units
F _{SMB}	SMBus Frequency	10		400	KHz
T _{BUF}	Time Between Stop and Start	1.44			μs
T _{HD,STA}	Hold Time After Start Condition. After This Period, the First Clock is Generated.	0.48			μs
T _{SU,STA}	Start Condition Setup Time	1.6			μs
T _{SU,STO}	Stop Condition Setup Time	1.76			μs
T _{HD,DAT}	Data in Hold Time	0.32			μs
T _{SU,DAT}	Data in Setup Time	0.1			μs
T _{LOW}	SMBCLK Low Time	0.8			μs
T _{HIGH}	SMBCLK High Time	1.44			μs

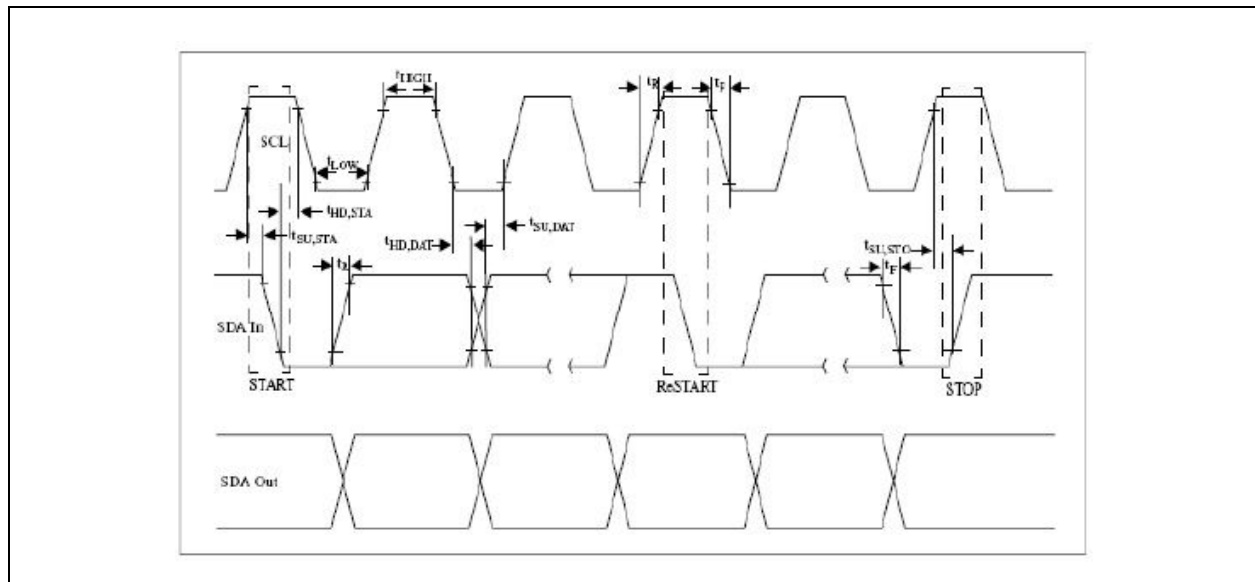


Figure 13-4. SMBus I/F Timing Diagram



13.6.2.3 Flash AC Specification

The X710-TM4 is designed to support a serial Flash applicable over a recommended operating range from $T_a = 0\text{ }^{\circ}\text{C}$ to $+70\text{ }^{\circ}\text{C}$, $V_{CC3P3} = 3.3\text{V}$, $C_{load} = 16\text{ pF}$ (unless otherwise noted). For Flash I/F timing specifications, see [Table 13-24](#) and [Figure 13-5](#).

This section applies to FLSH_SI, FLSH_SO, FLSH_SCK and FLSH_CE_N.

Table 13-24. Flash I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{SCK}	FLSH_SCK Clock Frequency	0		25	MHz	1
t_{RI}	FLSH_SO Rise Time		2.5	20	ns	
t_{FI}	FLSH_SO Fall Time		2.5	20	ns	
t_{WH}	FLSH_SCK High Time	20	50		ns	2
t_{WL}	FLSH_SCK Low Time	20	50		ns	2
t_{CS}	FLSH_CE_N High Time	25			ns	
t_{CSS}	FLSH_CE_N Setup Time	15			ns	3
t_{CSH}	FLSH_CE_N Hold Time	25			ns	
t_{SU}	Data-in Setup Time	5			ns	
t_H	Data-in Hold Time	5			ns	
t_V	Output Valid			20	ns	
t_{HO}	Output Hold Time	0			ns	
t_{DIS}	Output Disable Time			100	ns	
t_{EC}	Erase Cycle Time per Sector			1.1	Seconds	
t_{BPC}	Byte Program Cycle Time		60	100	μs	

1. Clock is either 25 MHz or 26.04 MHz divided by 2.
2. 50% duty cycle.
3. Timing is measured at 10% - 90%.

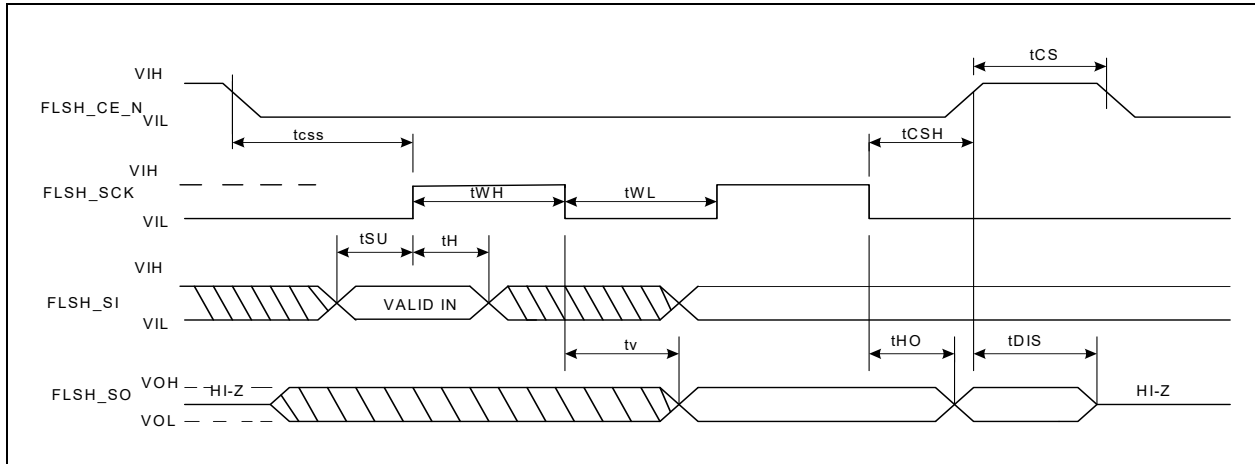


Figure 13-5. Flash I/F Timing Diagram

13.6.2.4 NC-SI AC Specifications

The X710-TM4 supports the NC-SI standard as defined in the DMTF Network Controller Sideband Interface (NC_SI) specification. The NC-SI timing specifications can be found in Table 13-25 and Figure 13-6.

Table 13-25. NC-SI Interface AC Specifications

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Units	Notes
REF_CLK Frequency				50	50+100 ppm	MHz	
REF_CLK Duty Cycle			35		65	%	2
Clock-to-Out (10pF<=load<=50 pF)	Tco		2.5		12.5	ns	1,3
Skew Between Clocks	Tskew				1.5	ns	
TXD[1:0], TX_EN, RXD[1:0] and CRS_DV Data Setup to REF_CLK Rising Edge	Tsu		3			ns	3
TXD[1:0], TX_EN, RXD[1:0] and CRS_DV Data Hold From REF_CLK Rising Edge	Thd		1			ns	3
Signal Rise/Fall Time	Tr/Tf		1		6	ns	4
REF_CLK Rise/Fall Time	Tckr/Tckf		0.5		3.5	ns	5
Interface Power-Up High Impedance Interval	Tpwrz		2			µs	
Power Up Transient Interval (recommendation)	Tpwrt				100	ns	
Power Up Transient Level (recommendation)	Vpwrt		-200		200	mV	



Table 13-25. NC-SI Interface AC Specifications (Continued)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Units	Notes
Interface Power-Up Output Enable Interval	Tpwre				10	ms	
EXT_CLK Startup Interval	Tclkstrt				100	ms	

Notes:

1. This timing relates to the output pins timing while T_{su} and T_{hd} relate to timing at the input pins.
2. REF_CLK duty cycle measurements are made from V_{ckm} to V_{ckm} . Clock skew T_{skew} is measured from V_{ckm} to V_{ckm} of two NC-SI devices and represents maximum clock skew between any two devices in the system.
3. All timing measurements are made between V_{ckm} and V_m . All output timing parameters are measured with a capacitive load between 10 pF and 50 pF.
4. Rise and fall times are measured between points that cross 10% and 90% of V_{ref} (see Table 13-14). The middle points (50% of V_{ref}) are marked as V_{ckm} and V_m for clock and data, respectively.
5. A serial 33Ω resistor might be required in case of a very short trace on the board.

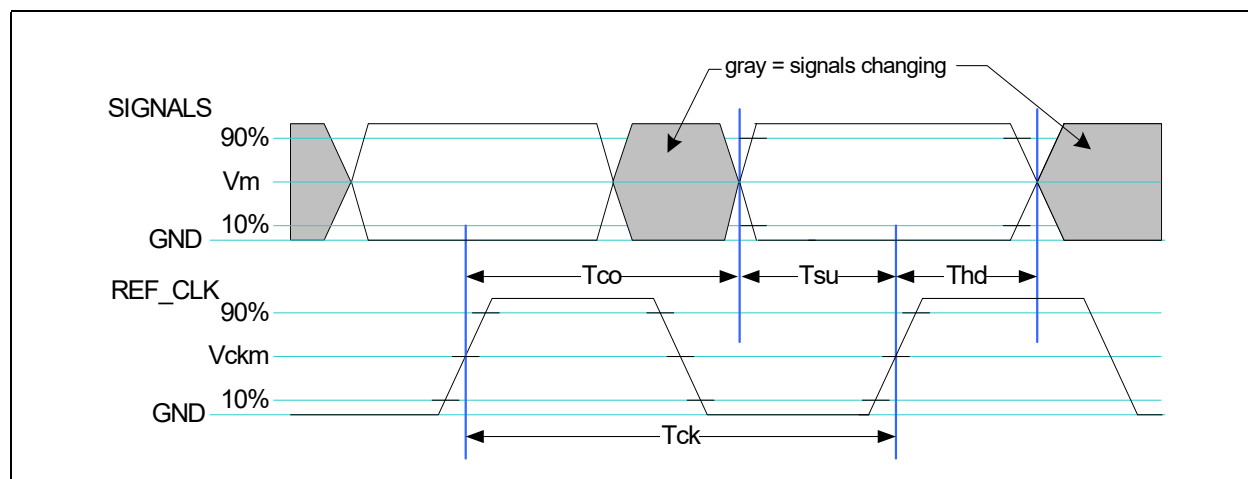


Figure 13-6. NC-SI AC Timing Diagram

13.6.2.5 JTAG AC Specification

The X710-TM4 is designed to support the IEEE 1149.1 standard. The following timing specifications are applicable over a recommended operating range from $T_a = 0\text{ }^\circ\text{C}$ to $+70\text{ }^\circ\text{C}$, $V_{CC3P3} = 3.3\text{V}$, $C_{load} = 16\text{ pF}$ (unless otherwise noted). For JTAG I/F timing specifications, see Table 13-26 and Figure 13-7.

This section applies to JTCK, JTMS, JTDI and JTDO. Also, timing is measured relative to the JTCK reference voltage of $V_{CC3P3}/2$.

Table 13-26. JTAG I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{JCLK}	JTCK Clock Frequency			10	MHz	
t_{JH}	JTMS and JTDI Hold Time	10			ns	
t_{JSU}	JTMS and JTDI Setup Time	10			ns	
t_{JPR}	JTDO Propagation Delay			15	ns	

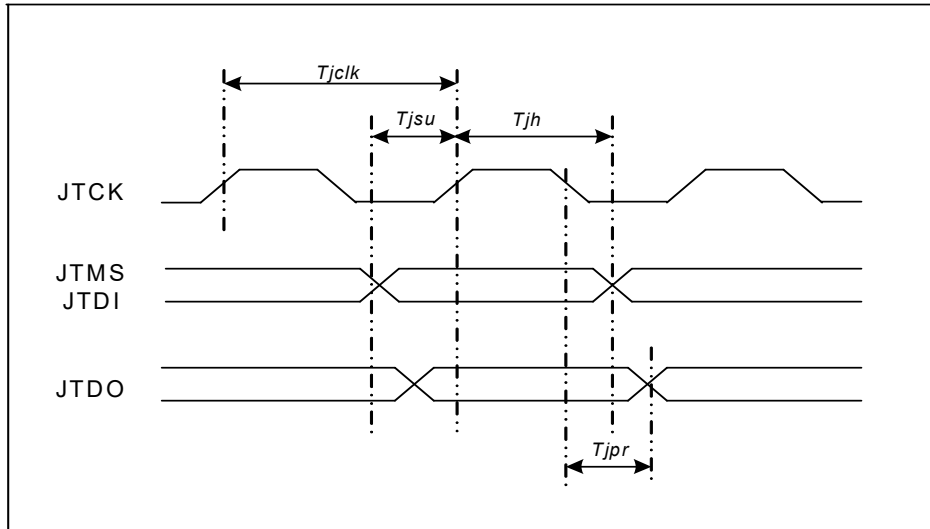


Figure 13-7. JTAG AC Timing Diagram

13.6.2.6 MDIO AC Specification

The X710-TM4 is designed to support the MDIO specifications defined in IEEE Std. 802.3-2018 clause 22. The following timing specifications are applicable over a recommended operating range from $T_a = 0^\circ\text{C}$ to $+70^\circ\text{C}$, $V_{CC3P3} = 3.3\text{V}$, $C_{load} = 16\text{ pF}$ (unless otherwise noted). For MDIO I/F timing specifications, see Table 13-27, Figure 13-8 and Figure 13-9.

This section applies to MDIO0, MDC0, MDIO1 and MDC1. Also, timing is measured relative to the MDC reference voltage of 2.0V (V_{ih}).

Table 13-27. MDIO I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{MCLK}	MDC Clock Frequency	2.4		24	MHz	
t_{MH}	MDIO Hold Time	10			ns	
t_{MSU}	MDIO Setup Time	10			ns	
t_{MPR}	MDIO Propagation Delay	10		30	ns	

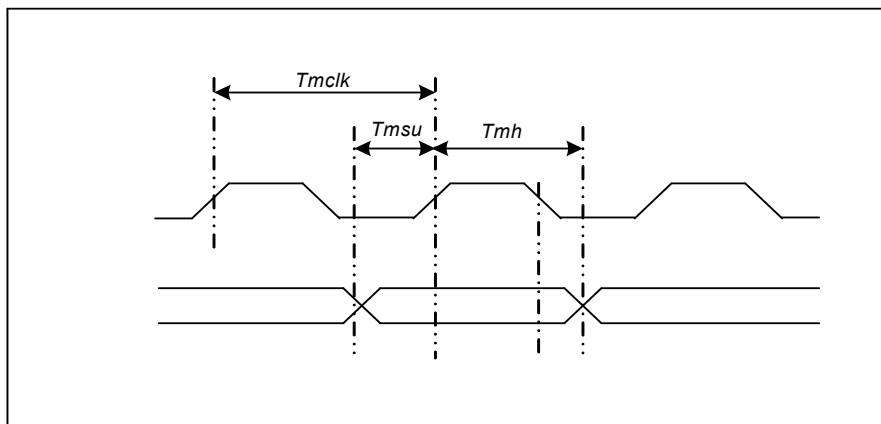


Figure 13-8. MDIO Input AC Timing Diagram

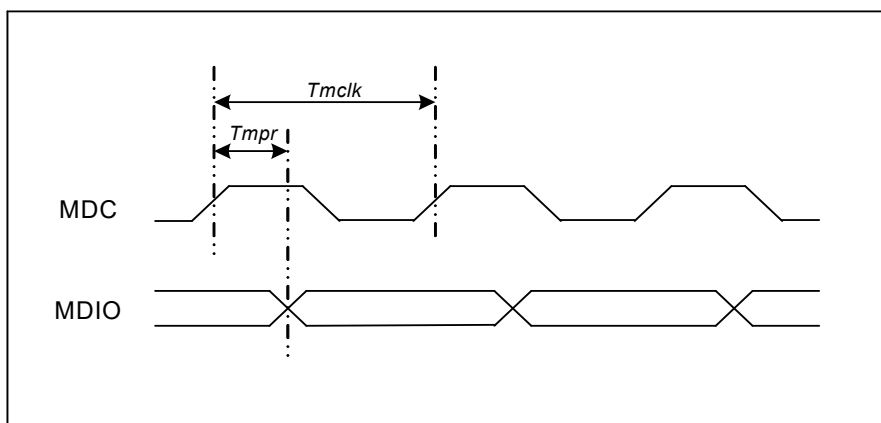


Figure 13-9. MDIO Output AC Timing Diagram

13.6.2.7 Reset Signals

For a power-on indication, the X710-TM4 can either use an internal power-on circuit, which monitors the VCCD power supply, or an external reset using the LAN_PWR_GOOD pin. The POR_BYPASS pin defines the reset source (when high, the device uses the LAN_PWR_GOOD pad as power-on indication). By default, internal POR should be used. Also, the BASE-T ports 0/1 need a separate hardware reset prior to normal operation.

Note: The timing between the power-up sequence and the different reset signals when using the internal power indication is described in [Section 13.3.1.1](#).

BYPASS mode is described in [Section 13.6.2.7.1](#).

A schema of the power-on logic is shown in [Figure 13-10](#).

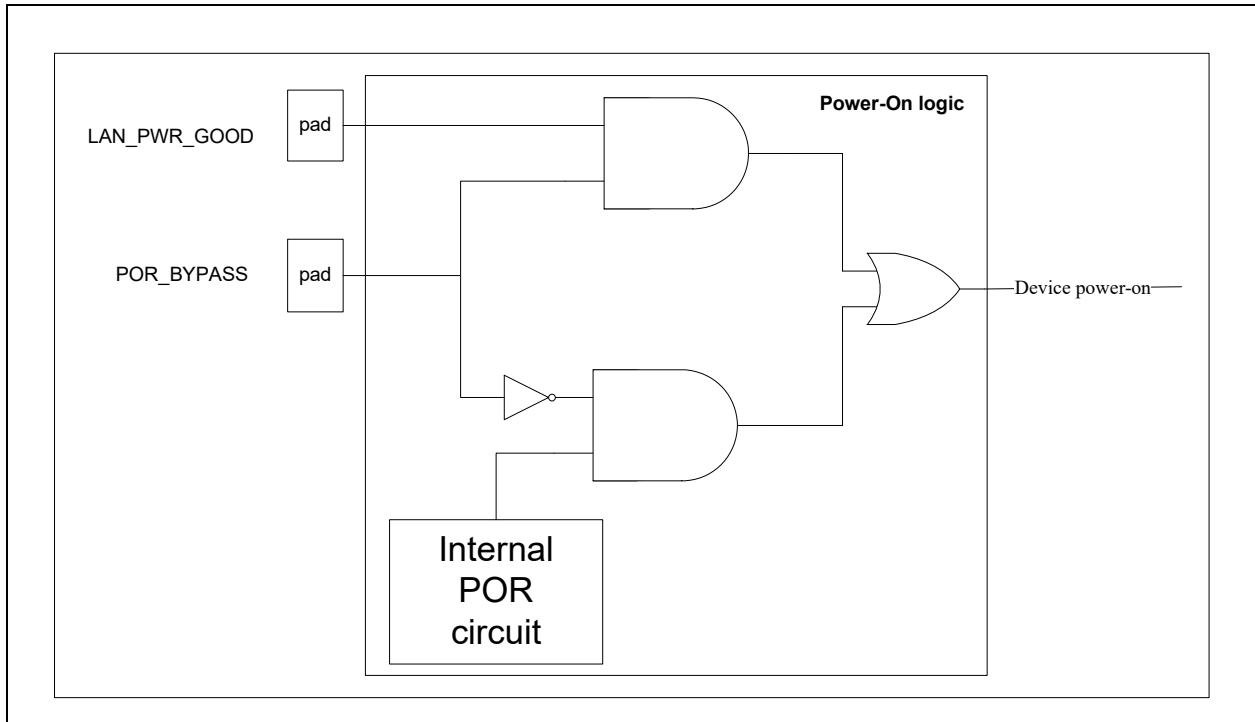


Figure 13-10. Schema of Power-on Logic

13.6.2.7.1 Power-on Reset Bypass

When asserting the POR_BYPASS pad, the X710-TM4 uses the LAN_PWR_GOOD pin as power-on indication. Otherwise, the X710-TM4 uses an internal power on detection circuit in order to generate the internal power on reset signal (see [Figure 13-11](#)).

Table 13-28 lists the timing for the external power-on signal:

Table 13-28. External Reset Specification

Symbol	Title	Description	Min	Max	Units
Tlpgw	LAN_PWR_GOOD minimum width	Minimum width for LAN_PWR_GOOD de-assertion	40	N/A	mS
Tpwr_lpg	Power-ramp to LAN_PWR_GOOD	Time from power-up of all power-rails to assertion of LAN_PWR_GOOD	40	N/A	mS
Tlpg_prst	LAN_PWR_GOOD to PE_RST_N	Time from LAN_PWR_GOOD assertion to PCIe reset de-assertion	51	N/A	mS
Tpwr_prst	Power-Ramp to PE_RST_N	Time from power-up of all power-rails to de-assertion of PE_RST_N	100		mS

The LAN_PWR_GOOD and POR_BYPASS are 3.3V (LVTTL) digital I/Os and their characteristics are described in [Section 13.6.1](#).

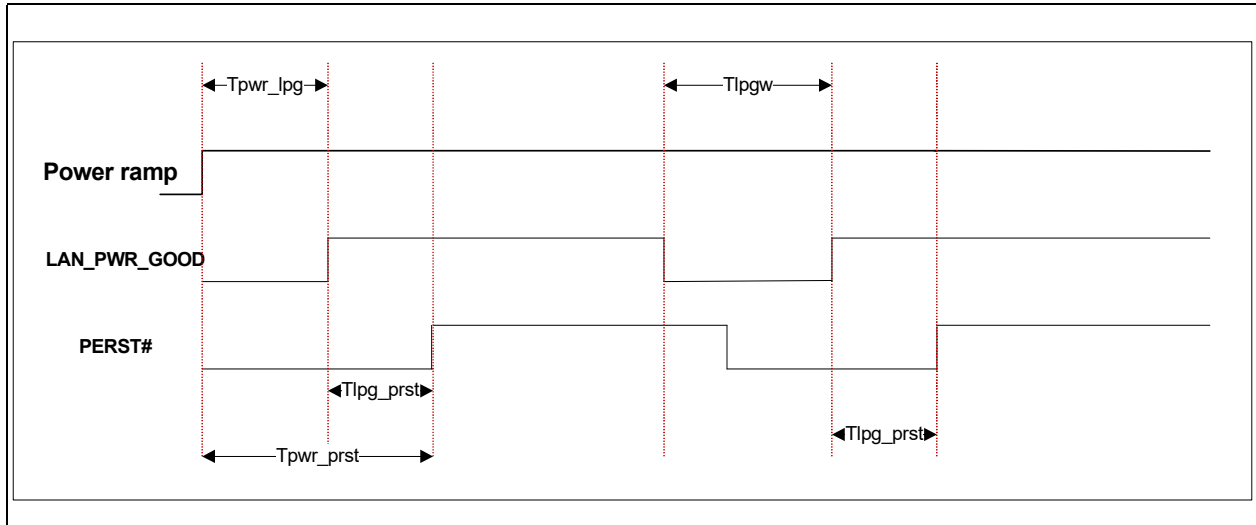


Figure 13-11. LAN_PWR_GOOD Timing

13.6.3 PCIe Interface AC/DC Specification

The X710-TM4 PCIe interface supports the PCIe Gen3.0 electrical specification defined in the:

- PCIe Express Base Specification Revision 3.0
- PCI Express Card Electromechanical Specification, Revision 3.0

13.6.4 Network (MAUI) Interface AC/DC Specification

13.6.4.1 KR Interface AC/DC Specification

The X710-TM4 MAUI interface supports the 10GBASE-KR electrical specification defined in IEEE Std. 802.3ap clause 72.

13.6.4.2 SFI+ Interface AC/DC Specification

The X710-TM4 MAUI interface supports the SFI electrical specification defined in the SFI+ MSA (SFF Committee SFF-8431).

13.6.4.3 BASE-T Interface AC/DC Specification

The X710-TM4 supports 100BASE-T, 1000BASE-T, 2.5GBASE-T, 5GBASE-T, and 10GBASE-T as defined by the IEEE Std. 802.3-2018 specification.

13.6.5 Crystal/Reference Clock Specification

13.6.5.1 25 MHz Crystal Specification

Figure 13-12 shows the usage of the crystal. This scheme is not a part of the X710-TM4 specification but rather an example that meets the required specification.

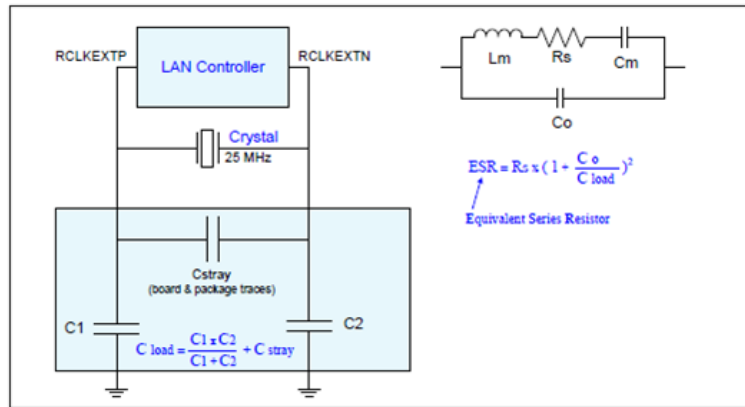


Figure 13-12. Crystal Usage

Table 13-29. Crystal Specifications

Parameter Name	Symbol	Recommended Value	Conditions
Frequency	f_o	25 MHz	@25 [°C]
Vibration Mode		Fundamental	
Cut		AT	
Operating /Calibration Mode		Parallel	
Frequency Tolerance @25 °C	Df/f_o @25 °C	±30 [ppm]	@25 [°C]
Temperature Tolerance	Df/f_o	±30 [ppm]	
Operating Temperature	T_{opr}	-20 to +70 [°C]	
Non Operating Temperature Range	T_{opr}	-40 to +90 [°C]	
Equivalent Series Resistance (ESR)	ESR	50 [Ω] maximum	@25 [MHz]
Load Capacitance	C_{load}	20 [pF]	
Shunt Capacitance	C_o	6 [pF] maximum	
Pull-capable From a Nominal Load Capacitance	Df/C_{load}	15 [ppm/pF] maximum	
Max Drive Level	D_L	0.5 [mW]	
Insulation Resistance	IR	500 [MΩ] minimum	@ 100V DC



Table 13-29. Crystal Specifications

Parameter Name	Symbol	Recommended Value	Conditions
Aging	Df/f_0	±5 [ppm/year]	
External Capacitors	C_1, C_2	20 [pF]	
Board Resistance	R_s	0.1 [Ω]	

Table 13-30. BASE-T PHY Crystal Specifications

Parameter Name	Min	Typical	Max	Units	Notes
Nominal Frequency	50			MHz	
Oscillation Mode	Fundamental				
Load Capacitance	12			pF	
Frequency Tolerance	±10			ppm	Measured at @25 °C ±3 °C
Frequency Stability	±25			ppm	Over an operating temperature range (reference 25 °C)
Operating Temperature	-10		+70	°C	
Aging	±2			ppm	First year
Drive Level	100		200	μW	
Effective Resistance R_r			15	Ω	
Shunt Capacity C_0		0.83		pF	
Motional Capacitance		2.97		fF	
Motional Inductance		3.41		mH	
Insulation Resistance	500			M Ω	Measured at DC 100V
Storage Temperature Range	-40		+85	°C	

13.6.5.2 25 MHz Differential Clock Generator Specification

Figure 13-13 shows an example for using external oscillator (PECL or CML). This scheme is not part of the X710-TM4 specification but rather an example that meets the required specification. However, complete validation was made only using a PE77D042-25.0M (PECL clock oscillator).

Note: An external oscillator must have a differential output. The X710-TM4 does not operate with a single-ended oscillator. Also, routing the input clock between the external oscillator and the X710-TM4 balls must be routed as a differential pair; target 100 Ω differential impedance.

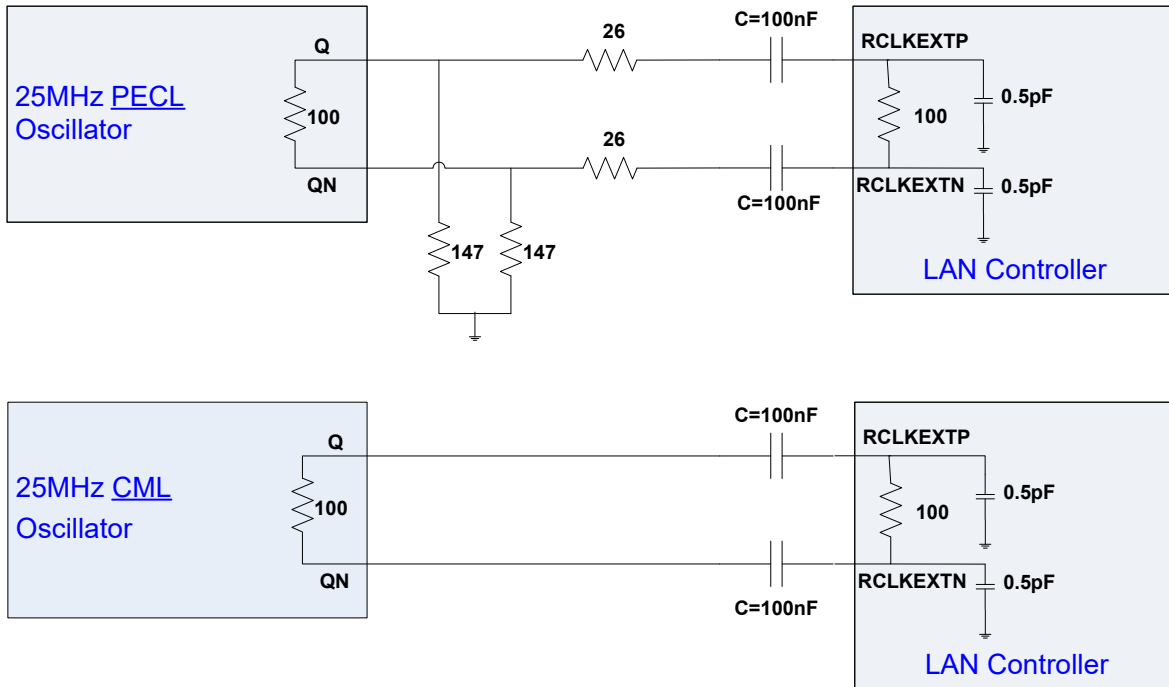


Figure 13-13. External Oscillator Usage (PECL or CML)

Table 13-31. Input Reference Clock Electrical Characteristics

Sym	Parameter	Min	Typ	Max	Unit	Comments
f	Frequency		25		MHz	
Δf	Frequency Variation	-100		+100	ppm	
DC	Duty Cycle	45		55	%	
Tr	Rise Time (20% - 80%)	50		1000	ps	
Tf	Fall Time (20% - 80%)	50		1000	ps	
AMP	Differential Peak-to-Peak Amplitude	0.6		1.25	V	
C	AC Coupling		100		nF	
Cin	Input Capacitance		0.5		pF	
DJ P2P	Deterministic Peak-to-Peak Jitter			10	ps	
p-noise	Phase Noise		-145		dBc/Hz	

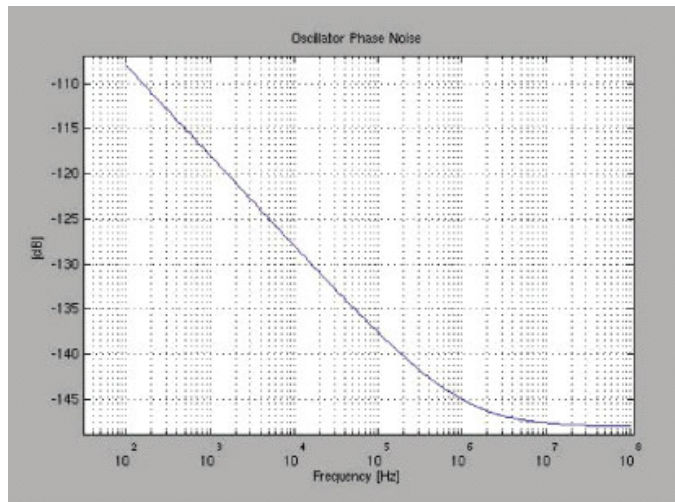
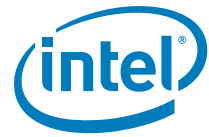


Figure 13-14. Maximum External Oscillator Phase Noise as a Function of Frequency (High-speed Serial)

13.6.6 BIAS Resistor Connection

The X710-TM4 uses a single bias circuit common to both PCIe and MAUI analog blocks.

To properly bias the high speed analog interfaces of the X710-TM4, a 4.75 KΩ 0.1% resistor needs to be connected between the RBIAS (ball Y10) to RSENSE (ball Y12). The voltage drop measurable on this bias resistor is 950 mV.

13.7 Package

13.7.1 Mechanical

The X710-TM4 is assembled in a 22 x 22 FCGBA package with an 8-layer substrate.

Table 13-32. Package Specifications

Body Size	Ball Count	Ball Pitch	Ball Matrix	Substrate
22 x 22 mm	503	1 mm		TBD



13.7.2 Thermal

Thermal models are intended to assist OEMs in developing system-level thermal solutions. It includes a FloTherm simulation mode. Refer to the Intel® Ethernet X710-TM4 FloTherm Model for more detail.

Note: The mounting pressure for any heat sink solution is 10 lbf or 44.5 N.

13.7.3 Electrical

Package electrical models are part of the IBIS files.

13.8 Mechanical Package

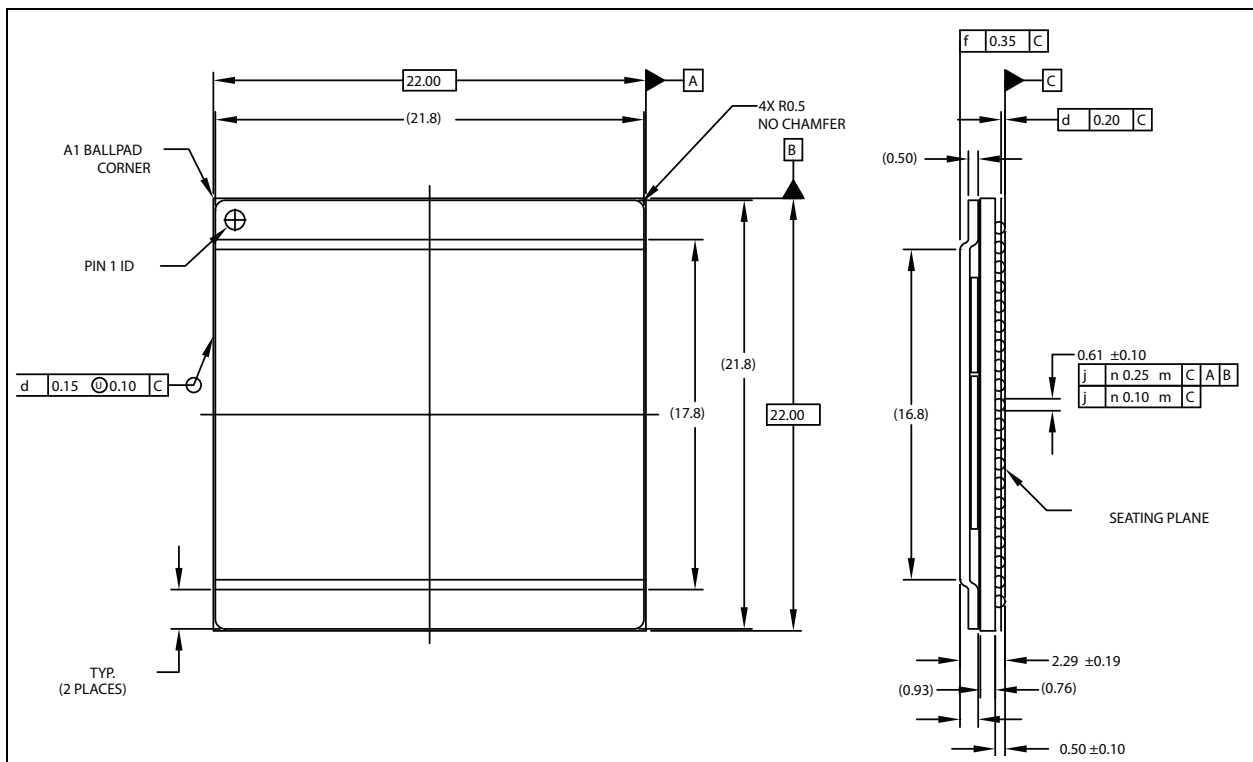
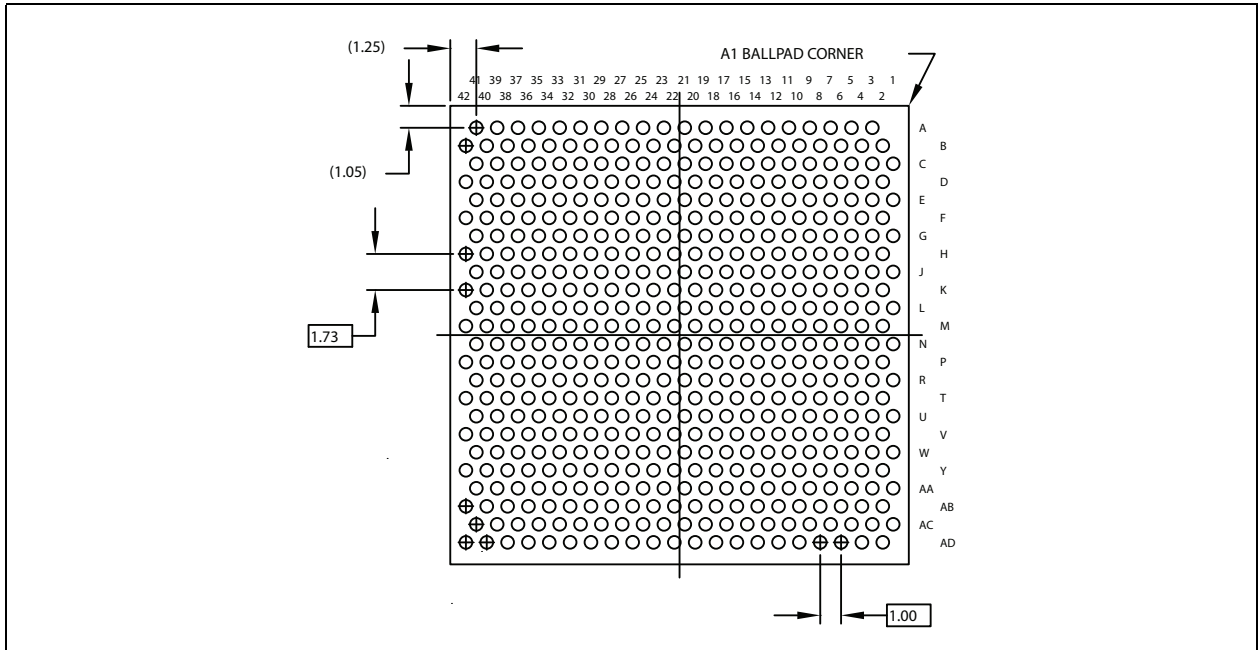


Figure 13-15. Mechanical Package Diagram (Top/Side View)



Mechanical Package Diagram (Bottom View)



NOTE: *This page intentionally left blank.*



14.0 Design Guidelines

14.1 Connection Considerations and Guidelines

This section provides explicit recommendations for selecting components, connecting interfaces, operating with special pins, thermal considerations and layout guidance to achieve a successful X710-TM4 design. The X710-TM4 has several features and interface configuration options that provide system designers the freedom to architect the design in a multitude of ways.

Note: Before implementing any X710-TM4 design based on the following guidelines, designers need to have the latest revision of this document. Also, it is strongly recommended that the design schematics and board layouts are reviewed by your Intel Field Service Representative before designs are taped out.

14.2 Naming Conventions

General naming conventions included in this section are as follows:

- Pins named as RSVDxx_NC are reserved and should be left unconnected.
- Pins named RSVDxx_VSS are reserved and should be pulled low to GND.

In addition to the previously mentioned pins needing pull-up or pull-down resistors, some interfaces, when not connected, should also be terminated with pull-up or pull-down resistors. Others must be left open. Unless otherwise specified, the recommended value of pull-up resistors ranges from 3.3 K Ω to 100 K Ω . Recommended pull-down resistor values ranges from 100 Ω to 800 Ω . Please refer to the interface descriptions provided in the sections that follow for more details.

Unless the strapping requirements are followed, the X710-TM4 can enter special test modes and present unexpected behavior.

14.3 PCIe Interface

The X710-TM4 connects to a host system using the PCIe interface. The interface can be configured into different modes of operation detailed in [Section 3.1.4](#).



When routing these differential signals, note that the channel needs to comply with signal integrity requirements of the Gen 1 (2.5GT/s), Gen 2 (5GT/s), Gen 3 (8GT/s) interface. For more detail, refer to the layout recommendations in [Section 14.14](#).

14.3.1 Link Width Configuration

The X710-TM4 supports link widths of x8, x4, or x1. The configuration is loaded from the NVM into bits 9:4 in the *Max Link Width* field of the Link Capabilities register (0xAC), the default link width being x8.

During link configuration, the upstream device and the X710-TM4 negotiate a common link width. In order for this to work, the selected maximum number of PCIe lanes must be physically connected to the host system.

14.3.2 Polarity Inversion and Lane Reversal

To ease routing, designers have the flexibility to use the lane reversal modes supported by the X710-TM4. Polarity inversion can also be used since the polarity of each differential pair is detected during the link training sequence.

Note: When lane reversal is used, some of the down-shift options are not available. For a description of available combinations, see [Section 3.1.4.3](#).

14.3.3 AC Coupling

Each PCIe lane has to be AC-coupled between its corresponding transmitter and receiver; with the AC-coupling capacitor located close to the transmitter side (within 1 inch).

Note: The recommended value of the AC coupling capacitors depends on the link speed.

For designs that support PCIe speeds up to Gen 1 (2.5GT/s) and Gen 2 (5GT/s), a value of 100 nF is recommended. For designs that include support for Gen 3 (8GT/s) speeds, a value of 220 nF is recommended as stated in the PCIe Base specification 3.0 for 5.0GT/s speeds or lower the capacitance value should be between 75 nF-265 nF. For 8GT/s speeds, the capacitance value should be between 176 nF to 265 nF.

14.3.4 PCIe Terminations

Each PCIe differential pair is terminated on-die; board termination is not required.

14.3.5 Miscellaneous PCIe Signals

The X710-TM4 signals power management events to the system by pulling the PE_WAKE_N signal (pin P2) signal low. Note that somewhere in the system, this signal must be pulled high to the auxiliary 3.3V supply rail.



The PE_RST_N signal (pin R1), which serves as the familiar reset function for the controller chip, needs to be connected to the host system's corresponding signal.

The PCIe interface also requires a reference clock that is usually provided by the host system. For more details, refer to [Section 14.7](#).

14.4 MAUI Interfaces

This section outlines the LAN interface configuration options and lane mapping guidelines for the X710-TM4. Choose the appropriate configuration for your system design.

The X710-TM4 offers a maximum of four 10 GbE ports: two 10 GbE BASE-T ports and two 10 GbE KR/SFI serial ports.

The interface categories supported in each of these cases are slightly different:

- In 10GBASE-T mode, supported interface options are 1/2.5/5/10GBASE-T and 100BASE-TX.
- In 10 GbE mode, supported interface options are KR or SFI.

The X710-TM4 PHY has four SerDes blocks, all of which support 10 GbE serial (KR/SFI) operation. See [Figure 14-1](#).

Note: SFP+ board traces generally do not require AC coupling capacitors because they are normally integrated into the pluggable SFP+ modules.

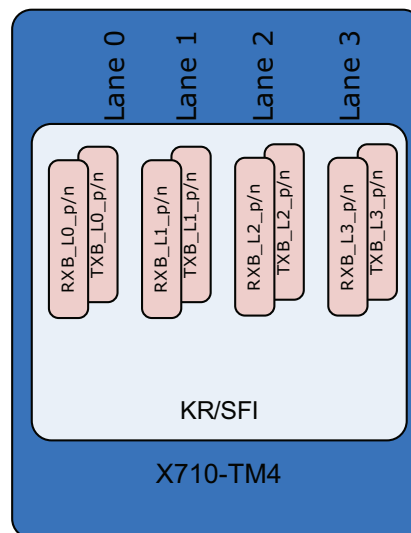


Figure 14-1. Lane Mapping

Figure 14-1 shows the lanes according to their physical location on the package when observed from above (die side).



14.4.1 Lane Mapping

The X710-TM4 ports can be assigned in many different lanes/lane combinations to enable implementation of backward-compatible designs or highly configurable designs (see [Figure 14-1](#)).

See [Table 3-31](#) for possible port-to-physical lane configurations allowed by the X710-TM4 for various PHY interfaces for single, dual and quad port configurations.

All SerDes differential lanes are 100 Ω terminated on die. Differential MAUI signals need to be AC coupled near the receiver. For recommended capacitor values, consult the relevant IEEE Std. 802.3-2018 specifications and/or the relevant PICMG specifications. Capacitor size should be small to reduce parasitic inductance. Use X5R or X7R, ±10% capacitors in a 0402 or 0201 package size.

14.5 Sideband Signals

This section describes the various sideband interfaces provided to operate with the internal PHY. Several of these sideband interfaces behave differently based on the use case. As such, the following section describes usage models like: 10GBASE-T, KR and SFI.

This section focuses on the various sideband interfaces provided to operate with the externally connected SFP+ modules. Sideband interfaces behave differently based on the use case. As such, the following section focuses on usage models like: SFP+ and 10GBASE-T.

14.5.1 Management Data Input/Output (MDIO) Interfaces

The X710-TM4 supports up to four MDIO interfaces (one per port) to control the PHY. The first two MDIO ports, MDIO0 and MDIO1, are at pin locations: MDC0_SCL0, MDIO0_SDA0 and MDC1_SCL1, MDIO1_SDA1 (pins N3, N1 and M4, M2). MDIO ports 2 and 3 are located at pins MDC2_SCL2, MDIO2_SDA2 and MDC3_SCL3, MDIO3_SDA3 pins are located at balls L3, L1 and K4, K2. All four MDIO ports can also be configured to operate in I²C mode as well. The operating mode of these interfaces is NVM configurable.

In order to manage the PHY, configure the MDIO interface of Port0 to control the PHY. The PHY configuration registers for each port are mapped into the respective MDIO address space and are accessible to the MAC or the MDIO controller firmware.

Table 14-1. X710-TM4 MDIO Port Mapping

Port 0	Port 1	Port 2	Port 3
MDC0_SCL0 MDIO0_SDA0	MDC1_SCL1 MDIO1_SDA	MDC2_SCL2 MDIO2_SDA2	MDC3_SCL3 MDIO3_SDA3

The X710-TM4 MDIO interfaces use 3.3 V LVTTTL signaling. As a result, interfacing external PHY devices using 1.8V signaling requires level translators.

When connecting the MDIO interfaces, make sure the appropriate MDIO and MDC signals are connected to the corresponding PHY pins.



14.5.2 I²C Interfaces

Since the X710-TM4 offers the option to implement a dual port SFP+ design, a total of two I²C interfaces are needed. To satisfy this requirement, the MDIO interfaces previously described have the flexibility to operate in I²C mode.

Table 14-2 lists the mapping of the I²C interfaces to the SFP+ ports:

Table 14-2. X710-TM4 I²C Interface to SFP+ Port Mapping

Port 2	Port 3
MDC2_SCL2, MDIO2_SDA2	MDC3_SCL3, MDIO3_SDA3

When implementing SFP+ interfaces, the I²C interfaces need to be connected to the I²C pins of the SFP+ connector. Please make sure to provide pull-up resistors to 3.3V on both SCL and SDA pins.

For more details, consult the X710-TM4 reference schematics.

14.6 General Purpose I/O (GPIO) Pins

The X710-TM4 has a total of 30 GPIO pins that can be configured as Software Definable Pins (SDPs), LED drivers, and dedicated hardware functions for connecting to the PHY or IEEE 1588 auxiliary devices. The GPIO pins can also be associated with any of the physical ports. The following sections show the default configuration for the GPIO pins reserved for specific use and are named by default as SDP, LED or GPIO signals. However, the X710-TM4 offers the flexibility to configure any of the GPIO pins, regardless of name, to different modes and associate them with different ports as described in [Section 3.5.3](#). It is not recommended to deviate from the default GPIO function allocation specified in this document in order to reduce the amount of support and extra validation needed.

14.6.1 Software-Definable Pins (SDPs)

By default, the X710-TM4 has a total of 16 SDPs. These pins can either operate in native mode or as a SDP. In native mode, they serve certain pre-defined functions and are used as sideband signals for interfacing SFP+ modules.

[Table 14-3](#) lists the native function of these SDPs in various scenarios.



Table 14-3. X710-TM4 Native SDP Functions

Port#	Pin Name	10GBASE-T/SFP+	10GBASE-T PHY
0	SDP0_0	PHY_RESET	PHY_RESET
	SDP0_1	PHY_INT	PHY_INT
	SDP0_2	PHY_INT	PHY_INT
	SDP0_3	TX_DISABLE	
1	SDP1_0	RX_LOS	
	SDP1_1	TX_FAULT	
	SDP1_2	MOD_ABS	PHY_INT
	SDP1_3	RS0/RS1	PHY_INT
2	SDP2_0		
	SDP2_1		
	SDP2_2		
	SDP2_3	TX_DISABLE	
3	SDP3_0	RX_LOS	
	SDP3_1	TX_FAULT	
	SDP3_2	MOD_ABS	
	SDP3_3	RS0/RS1	

Note: Table entries listed in gray are not SDPs but functionally belong to the sideband signals assigned to the same port.

Note: Note that the TX_DISABLE signal for the SFP+ implementation has been removed from the native SDP functions. This functionality is implemented through I²C register access. To support modules that don't support this feature, use one or more of the available GPIO pins. When connecting the SDPs to different digital signals, note that these are 3.3V signals and implement level shifting, if necessary.

The X710-TM4 enables all of these SDP pins to be configured as a general purpose interrupt pin. Interrupts can be generated on both the rising and falling edge of the signal. The rising or falling edge detection is implemented by comparing values sampled at the internal clock rate, as opposed to an edge-detection circuit.

14.6.2 Light Emitting Diodes (LEDs)

Each of the LED outputs can be individually configured to select the particular event, state, or activity indicated on that output. In addition, each LED can be individually configured for output polarity as well as for blinking versus non-blinking mode when asserted. For more information on the available states indicated or how to configure these pins from the EEPROM settings, see [Section 2.2.6.1](#).

By default, the X710-TM4 divides the LED pins and uses the same pin locations for LEDs. It divides them up differently based on the total number of ports supported by the implementation.

Note that the reduced number of LED pins in a four-port configuration presents some design limitations but the configuration flexibility of the GPIO pins allows for intuitive LED implementation.



To prevent damage to the interface and the LEDs themselves, provide separate current limiting resistors for each LED connected. Consider attaching filter capacitors to these outputs if the LEDs are placed close to the board edge and near noisy external interconnects that can cause EMI issues. Adequate separation in routing from noisy signals helps prevent such issues.

14.6.3 GPIO Pins

This section will be included in the next release of the datasheet.

14.6.4 MDIO/I²C/SDP/LED/GPIO Summary

Table 14-4 lists a summary of the various sideband interfaces and software definable pins (GPIOs configurable as LEDs, SDPs etc.). In the descriptions column of the quad-port implementations, each signal name is preceded by a prefix designating the port to which the signal in question belongs (such as P0-SCL).



Table 14-4. X710-TM4 MDIO/I²C/SDP/LED Summary

		Description (X710-TM4 Pin Functions in Several Use Cases)	
Pin Name (Default Function)	Ball	10GBASE-TSFP+	10GBASE-T
MDC0_SCL0	N3	MDC	MDC
MDIO0_SDA0	N1	MDIO	MDIO
MDC1_SCL1	M4	P1-SCL	MDC
MDIO1_SDA1	M2	P1-SDA	MDIO
MDC2_SCL2	L3		MDC
MDIO2_SDA2	L1		MDIO
MDC3_SCL3	K4	P3-SCL	MDC
MDIO3_SDA3	K2	P3-SDA	MDIO
SDP0_0	E5	PHY_Rst	PHY_Rst
SDP0_1	G5	PHY_Int	PHY_Int
SDP0_2	K6	PHY_Int	PHY_Int
SDP0_3	L5	TX_Disable	
SDP2_0	F6		
SDP2_1	J7		
SDP2_2	E7		
SDP2_3	G7	TX_Disable	
SDP1_0	R7	P1-RX_Loss	
SDP1_1	P4	P1-TX_Fault	
SDP1_2	P6	P1-Mod_Abs	
SDP1_3	H6	P1-RS0/RS1	
SDP3_0	N5	P3-RX_Loss	
SDP3_1	N7	P3-TX_Fault	
SDP3_2	M6	P3-Mod_Abs	
SDP3_3	M8	P3-RS0/RS1	
LED0_0	V4	LED0_0	LED0_0
LED0_1	V6	LED0_1	LED0_1
LED2_0	T4	LED2_0	LED2_0
LED2_1	T6	LED2_1	LED2_1
LED1_0	U5	LED1_01	LED1_0
LED1_1	U7	LED1_1	LED1_1
LED3_0	R3	LED3_0	LED3_0
LED3_1	R5	LED3_1	LED3_1
GPIO0	V40	GPIO3_LED_P0	GPIO3_LED_P0
GPIO1	W41	GPIO3_LED_P1	GPIO3_LED_P1
GPIO2	Y42	GPIO3_LED_P2	GPIO3_LED_P2
GPIO3	V42	GPIO3_LED_P3	GPIO3_LED_P3
GPIO4	H8		
GPIO5	L7	GPIO_POR_BYPASS	GPIO_POR_BYPASS



14.7 Reference Clocks

14.7.1 PCIe Reference Clock

The X710-TM4 requires a 100 MHz differential reference clock for the PCIe block. This signal is typically generated on the host system and routed to the PCIe port. For add-in cards, the clock is furnished at the PCIe connector.

This clock must have a frequency tolerance of ± 300 ppm and it must be connected to the PE_CLK_p and PE_CLK_n pins of the X710-TM4. The clock pins are AA1 and AB2 respectively. Designers have the flexibility to invert the polarity of this signal if that results in a cleaner routing of the board.

Note: Reference clock specifications are detailed in both the base and CEM specification. Consult both specifications to understand the full set of requirements. Refer to sections 4.3.7 (Base Specification) and 2.1.3 (CEM Specification) in particular.

14.7.2 MAUI Reference Clock

The X710-TM4 requires a reference clock that is being used to generate the high frequency clocks for the MAC and PHY blocks of the controller.

There are many oscillator solutions available. The clock source and distribution network should be designed to meet the necessary frequency and jitter requirements.

To configure the X710-TM4's PLLs correctly, the oscillator select pin OSC_SEL (pin J9) needs to be pulled high (for example, an 8.2K Ω resistor).

There are different oscillator solutions with their pros and cons as described in the sections that follow.

Table 14-5. Recommended Oscillators and Crystal

Component	Manufacturer	Part Number	Frequency
Crystal	TXC*	7M50070012	50 MHz
Oscillator	Silicon Labs* TXC	511ABA25M0000BAGR CA25000002	25 MHz 25 MHz

14.7.2.1 Fixed Crystal Oscillator

A packaged fixed crystal oscillator includes an inverter, a quartz crystal, and passive components. The device renders a consistent square wave output. Oscillators used with microprocessors are supplied in many configurations and tolerances. Crystal oscillators can be used in special situations (such as when shared clocking among devices). As clock routing can be difficult to accomplish, it is preferable to provide a separate clock source for each device by using clock buffers.

14.7.2.2 Programmable Crystal Oscillators

A programmable oscillator can be configured to operate at many frequencies. This device contains a crystal frequency reference and a Phase Lock Loop (PLL) clock generator. The frequency multipliers and divisors are controlled by programmable fuses.

PLLs are prone to exhibit frequency jitter. This contributes to the jitter of the transmitted signal even with the programmable oscillator working within its specified frequency tolerance. If the transmitted signal has high jitter and the receiver PLL's jitter tolerance is marginal, it can lose lock causing bit errors or link loss. Use of programmable oscillators is not generally recommended.

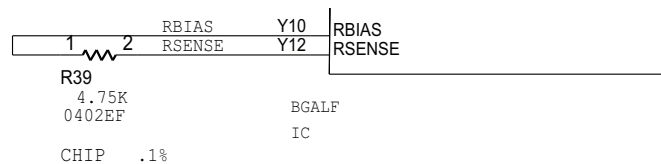
14.7.2.3 Reference Clock Measurement Recommendations

A low capacitance, high impedance probe ($C < 1 \text{ pF}$, $R > 500 \text{ K}$) should be used for testing. The measurement should be done on the XTAL_OUT (C3) pin. Incorrect probing setup or choice of probe affects the measurement and can lead to inaccurate results (from change in amplitude through clock frequency shift to even preventing the oscillations from starting).

14.8 Bias Resistors

The X710-TM4 uses a single bias circuit common to both PCIe and MAUI analog blocks. To properly bias the high speed analog interfaces, a 4.75 K Ω 0.1% resistor needs to be connected between the RBIAS (ball Y10) to RSENSE (ball Y12). The voltage drop measurable on this bias resistor is 950 mV.

To avoid noise coupled onto this reference signal, place the bias resistor close to the controller chip and keep traces as short as possible.



CONNECT A 4.75KOHM +/- 0.1% RESISTOR BETWEEN RBIAS AND RSENSE PINS
PLACE RESISTOR CLOSE TO CONTROLLER
TRACE LENGTH SHOULD BE < 1 INCH.

Figure 14-2. X710-TM4 Bias Circuit



14.9 NVM Devices

The X710-TM4 stores its configuration data in externally attached non-volatile memory chips, referred to as NVMs. Optional boot code packages are also stored in these externally attached non-volatile memory chips. These Flash devices are connected through the different Serial Peripheral Interface (SPI) as described in the section that follows.

14.9.1 SPI Flash

The SPI interface is used to connect to Flash devices. Flash devices are used for storing both configuration data and firmware as well as for storing the optional boot code packages.

The pins used to connect the Flash devices are FLASH_CE_N, FLASH_SCK, FLASH_SI and FLASH_SO (balls R41, P40, P42, and T42). Please note that the FLASH_SI pin is an output pin and needs to be connected to the serial input of a Flash device and the FLASH_SO is an input pin and needs to be connected to the serial output of a Flash device. For more information and the electrical characteristics of these pins see [Section 13.6.2](#). Due to the Flash size requirements detailed in the sections that follow designers must make sure that both size requirements can be met by devices with the chosen footprint.

14.9.2 Supported Flash Devices

The X710-TM4 designs require a 64 Mb SPI Flash device with an address size of 24 bits.

Table 14-6. Flash Devices

Density	Atmel* PN	Winbond*	SST*	Numonyx*	Adesto Tech*	Macronix*
64 Mb	AT25DF641	W25Q64JV W25Q64JVSSIQ	SST25VF064C	M25PX64	AT25SF641-SUB-T	MX25L6433FM2I-08G
128 Mb					AT255F128A-SHBHD	
256 Mb						M25L25645G

For more information on how to properly attach a Flash device, follow the example provided in the X710-TM4 reference schematics.

14.10 Manageability Interfaces

The X710-TM4 supports the SMBus, NC-SI interfaces, and PCIe (see [Section 3.1](#)) for pass-through and configuration traffic between the Management Controller (MC) and the X710-TM4. See [Section 9.0](#), for more information about MC support.

14.10.1 SMBus

The SMBus interface is formed by the SMBCLK, SMBD and SMBALRT_N signals. Since the SMBus specification does not specify the fastest rise time of these clock and data signals, in order to reduce potential PCIe signal integrity issues, the pins dedicated for this interface have been located at R39, T40, and U41.

Optionally, in order to further reduce high frequency content that could potentially couple as noise to other adjacent signals in the package, series resistors (of up to 1.1 K Ω) should be used on the SMBus signals.

If the interface is not being used, the design should provide a pull-up resistor on all of these signals.

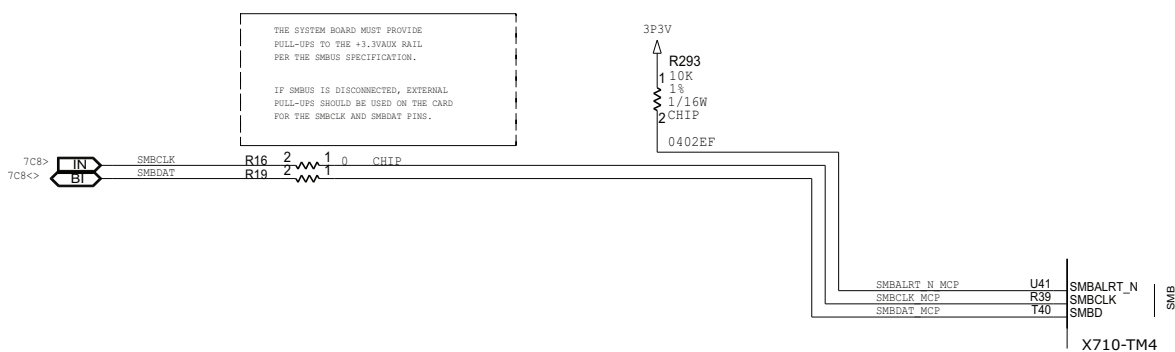


Figure 14-3. SMBus Connections

14.10.2 NC-SI

Management packets can be routed to or from an external management processor. The MC is required to meet the requirements called out in the DMTF NC-SI specification.

The X710-TM4 supports hardware arbitration therefore, two additional pins from the single point to point NC-SI solution, have been defined: NCSI_ARB_IN (pin G3) and NCSI_ARB_OUT (pin F4). These pins can be used to implement a hardware arbitration ring. The NCSI_ARB_IN pin of one controller has to be connected to the NCSI_ARB_OUT of another controller to form a ring configuration. A maximum of four network controllers can be connected in this manner and all controllers that share the same NC-SI interface pins must support this feature in order to use hardware-based arbitration

The following sections present examples of single- and multi-drop configurations, with pull-up and pull-down requirements for each.



14.10.2.1 Single-Drop Configuration

The simplest NC-SI configuration is a single drop configuration. This refers to a point-to-point connection between one MC and one network controller device. The following schematic shows an example. Since the hardware arbitration is not applicable for this scenario, the NCSI_ARB_IN and NCSI_ARB_OUT signals are not used. To enable correct operation, the NCSI_ARB_OUT should be looped back to the NCSI_ARB_IN pin of the network controller or hardware arbitration should be disabled.

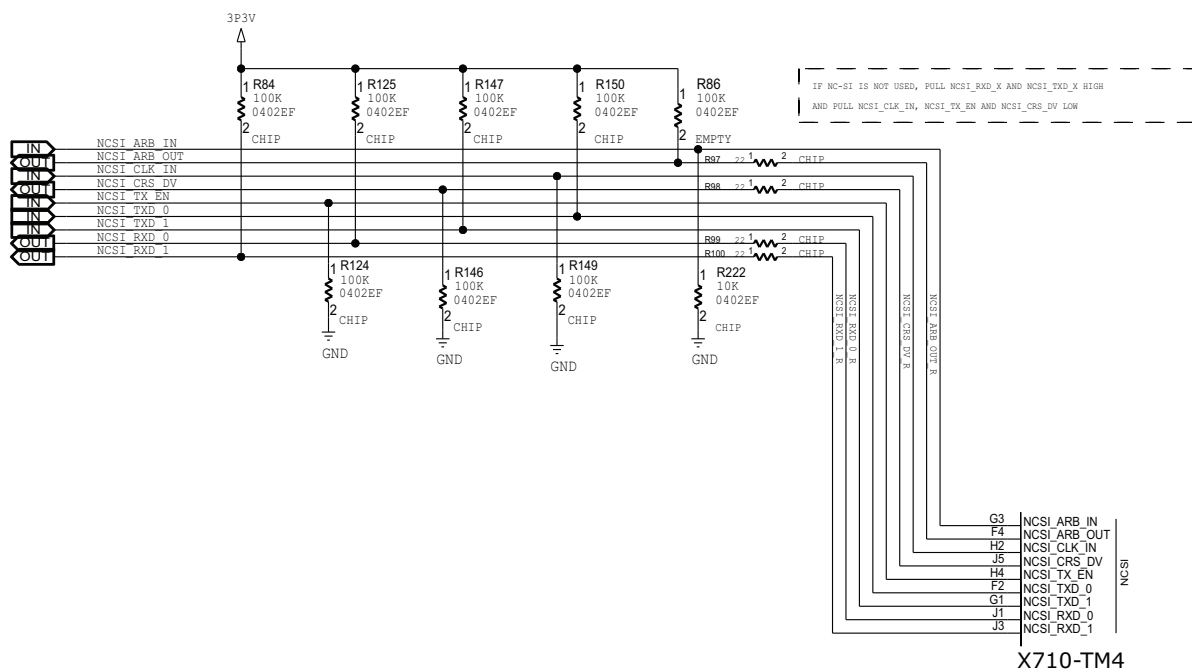


Figure 14-4. NC-SI Connection Schematic: Single-Drop Configuration

14.10.2.2 Multi-Drop Configuration

NC-SI architecture also enables multiple endpoints to be connected to the same management controller. In this configuration, the bus arbitration can also be implemented by hardware using a token ring configuration. The NCSI_ARB_IN pin of one controller must be connected to the NCSI_ARB_OUT of another controller to form a ring configuration.

A maximum of four network controllers can be connected in this manner and all controllers sharing the same NC-SI interface pins must support this feature in order to use hardware-based arbitration. See Section 2.2.3 for multi-drop schematic examples.

If the NC-SI interface is not used, pull the NCSI_CLK_IN, NCSI_CRD_DV, and NCSI_TX_EN signals low and the NCSI_RXD_0, NCSI_RXD_1, NCSI_TXD_0, and NCSI_TXD_1 signals high.



14.11 Miscellaneous Signals

14.11.1 Disable Signals

Two signals can be used for disabling Ethernet functions from system BIOS. Use pins PCI_DIS_N (pin U3) and DEV_DIS_N (D2). Both are latched at the rising edge of LAN_PWR_GOOD, PE_RST_N or In-Band PCIe Reset.

- **PCI_DIS_N:** If this pin is not connected or is driven high during initialization, then all PCI functions configured from the NVM are enabled. If this pin is driven low during initialization then all PCI functions that are allowed to be disabled as configured in NVM are disabled. When all bits in the corresponding NVM configuration word are set, all PCI functions are disabled when this pin is asserted low.
- **DEV_DIS_N:** If this pin is not connected or is driven high during initialization, all LAN ports configured from NVM are enabled. If this pin is driven low during initialization all LAN ports allowed to be disabled as configured in NVM are disabled. When all bits in the corresponding NVM configuration word are set, all LAN ports are disabled when this pin is asserted low. When a LAN port is disabled, the associated PCI functions are disabled as well.

The PCI_DIS_N and DEV_DIS_N pins should maintain their state during system reset and system sleep states as well as ensure the proper default value on system power-up. For example, a designer could use pin that defaults to 1b (enable) and is on system suspend power, meaning that it maintains it's state in S0-S5 ACPI states.

14.11.2 Power-on Reset (POR)

After power is applied, the X710-TM4 must reset. There are two ways to do this:

- Using the internal power on reset circuit (recommended).
- Using the external LAN_PWR_GOOD signal.

By default, the internal power on reset should be used. In this case, the power supply sequencing timing requirement between the 3.3V and VCCD (for the X710-TM4) power rails must be met. If this requirement is impossible to meet, the alternative is to bypass the internal power-on reset circuit by pulling POR_BYPASS (pin U39) high and using an external power monitoring solution to provide a LAN_PWR_GOOD signal (pin AB42).



Table 14-7. Reset Context for POR_BYPASS and LAN_PWR_GOOD

POR_BYPASS	Active Reset Circuit	
If = 0b	Internal POR	
If = 1b	LAN_PWR_GOOD	External Reset
	If = 0b	Held in reset.
	If = 1b	Initialized, ready for normal operation.

It is important to ensure that resets for the MC and the X710-TM4 are generated within a specific time interval. The important requirement here is to ensure that the NC-SI link is established within two seconds of the MC receiving the power-good signal from the platform. Both the X710-TM4 and the external MC need to receive power-good signals from the platform within one second of each other.

The MC should poll this interface and establish a link for two seconds to ensure specification compliance.

14.11.3 POR Without a Power Cycle for the X710-TM4 NVM Update

A circuit added to a X710-TM4 layout enables the NVM update tool to load several NVM modules into the X710-TM4 without a power cycle. See Table 6-2. The circuit simulates an A/C power cycle to the X710-TM4 by asserting an internal POR. For this circuit to function it requires enabling the selected GPIO as an output in the NVM image and a revised NVM update tool configuration file.

The standard NVM update flow requires a X710-TM4 power cycle and reboot to load several NVM modules into the X710-TM4 and bring it to a ready state. The POR_BYPASS circuit shown in Figure 14-5 and associated NVM update tool avoids a X710-TM4 power cycle.

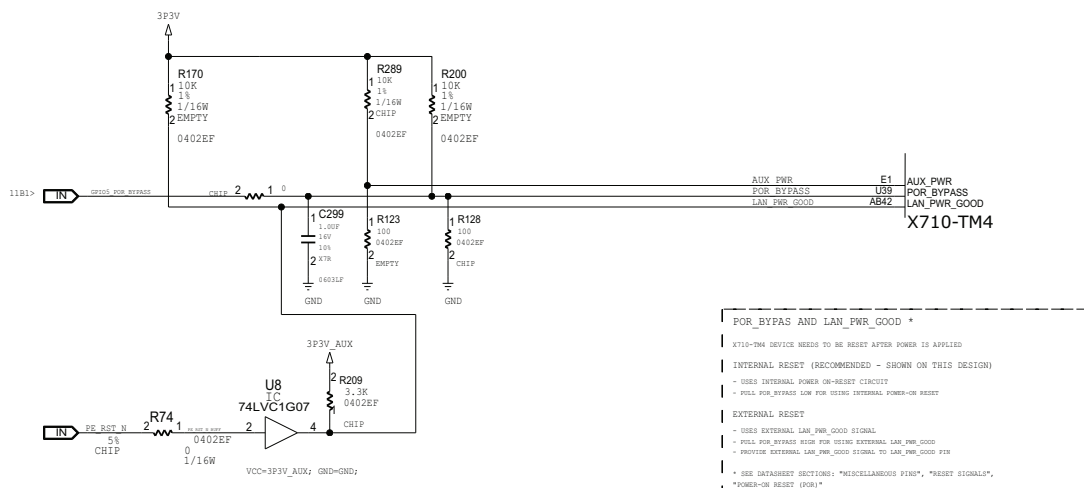


Figure 14-5. POR_BYPASS RC Circuit



The NVM update tool generates the X710-TM4 POR without a power cycle when the POR_BYPASS circuitry, NVM image configuration and revised NVM update configuration file are present. For POR without a power cycle:

1. The NVM update tool asserts the GPIO{n} signal, (n = 0 -- 5).
2. While POR_BYPASS is asserted, users initiate a system reboot after running the NVM update tool, causing a PE_RST_N.
3. PE_RST_N toggles the LAN_PWR_GOOD signal, initializing the X710-TM4.

Although the GPIO pin is no longer driven, the RC circuit keeps it high long enough for the reset to be recognized.

4. Two reboots are required to load all the new NVM modules into the X710-TM4 and brings it to a ready state.

Alternatively, system designers can avoid the two reboots through one of the following two cases:

- a. Ensure that PE_RST_N is asserted for 85 ms or longer on every reboot.
- b. Ensure that a double PE_RST_N happens on every reboot with a minimum of 85 ms delay between the start of first PE_RST_N and the end of the second.

14.11.3.1 NVM Update Tool Details

By default, the NVM update tool performs the standard update process. The NVM update configuration file must be revised to generate the POR without a power cycle as follows:

1. When the NVM update configuration file keyword FEATURES contains DOUBLE_REBOOT_GPIO {n}, (n = 0 -- 5), the NVM update tool invokes the POR without a power cycle process. Note that the NVM must properly enable the GPIO signal in the NVM.
2. The NVM update tool compares the device PHY and PCIe module CSS signatures to the NVM PHY and PCIe module CSS signatures. If one of the CSS signatures differs, the NVM update tool asserts the specified GPIO{n}.
3. The following message is output after the update:

A double Reboot is needed to complete the NVM update process.

Note: This message displays regardless of PE_RST_N duration and double reboot necessity.

Example CFG File using GPIO 5

```
CURRENT FAMILY: 12.12.12

CONFIG VERSION: 1.8.0

BEGIN DEVICE

DEVICENAME: X710-TM4

VENDOR: 8086

DEVICE: 1572

SUBVENDOR: 8086

SUBDEVICE: 0

FEATURES: DOUBLE_REBOOT_5

NVM IMAGE: NVM_image.bin

END DEVICE
```



14.11.3.2 Circuit Details

The following steps relate to the circuit in [Figure 14-5](#):

1. Pull down POR_BYPASS through a 1.5 K Ω resistor.
2. Connect one GPIO{*n*}, (*n* = 0 -- 5), to POR_BYPASS. Note that the NVM image default setting is GPIO[5] when the POR_BYPASS from GPIO feature is enabled.
3. Add a 1.0 μ F capacitor between POR_BYPASS to ground, parallel to the pull-down resistor.
4. Connect PE_RST_N through a buffer/driver to LAN_PWR_GOOD.

For additional information, contact your Intel representative.

14.11.4 Connecting the JTAG Port

The X710-TM4 offers a test access port conforming to the IEEE 1149.1-2001 Edition (JTAG) specification. The JTAG interface operates at 3.3V only. To use the test access port, connect the JTCK, JTMS, JTDI, JTDO and JRST_N pins (balls U37, V38, T36, T38, and V36) to test points accessible by appropriate test equipment.

For proper operation, a pull-down resistor with a value in the range of 470 Ω to 8.2 K Ω range should be connected to the JTCK and JRST_N signals and pull-up resistors with values in the K Ω range to the JTDO, JTMS and JTDI signals.

A Boundary Scan Definition Language (BSDL) file describing the X710-TM4 is currently available for use in your test environment.

All designs supporting the X710-TM4 must provide stuffing options to individually strap pins Y16, W7 and Y20 high or low to configure debug capabilities. All designs must also provide accessible test point connections to the JTAG interface to enable system debug.

Note: Intel strongly recommends that each X710-TM4 design provides a header for the JTAG interface to enable quick system debug. If the X710-TM4 is part of a JTAG chain the design should also provide stuffing options to isolate it from the chain. [Figure 14-6](#) shows the required connector (Samtec* TMM-103-01-L-D-SM or equivalent) and pin-out:

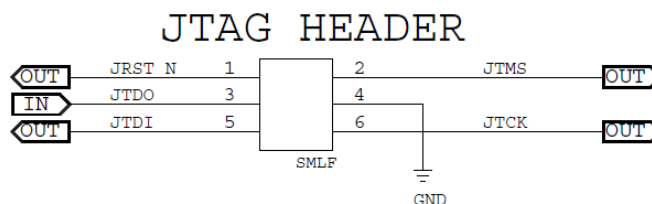


Figure 14-6. JTAG Header

Provide an external 1.0 K Ω resistor through unpopulated jumper to 3.3V on pins Y16, W7 and Y20. Provide an unpopulated jumper to optionally pull these signals up. Refer to the *Intel® Ethernet Controller X710-TM4 Reference Schematics* for more detail. Note that this jumper should only be used for debug purposes.

14.12 Power Supplies

The X710-TM4 requires the following power rails: 3.3V for the I/O ring, VCCA for the analog core and VCCD for the digital core. To provide a tight control over voltage levels and improve signal integrity, the VCCA rail is generated by a small on-die switching voltage regulator and is in the 0.9V - 1.0V range.

For powering the VCCD rail, designers need to provide an external switching voltage regulator. The voltage level needs to be adjusted based on process corner to optimize power consumption. The X710-TM4 provides additional pins for on-die voltage sensing and the necessary margining circuit.

The 3.3V rail needs to be externally derived from the system's main and/or auxiliary voltages.

14.12.1 VCCA Analog Supply

Figure 14-7 shows an example of the necessary connections to implement the on-die regulator.

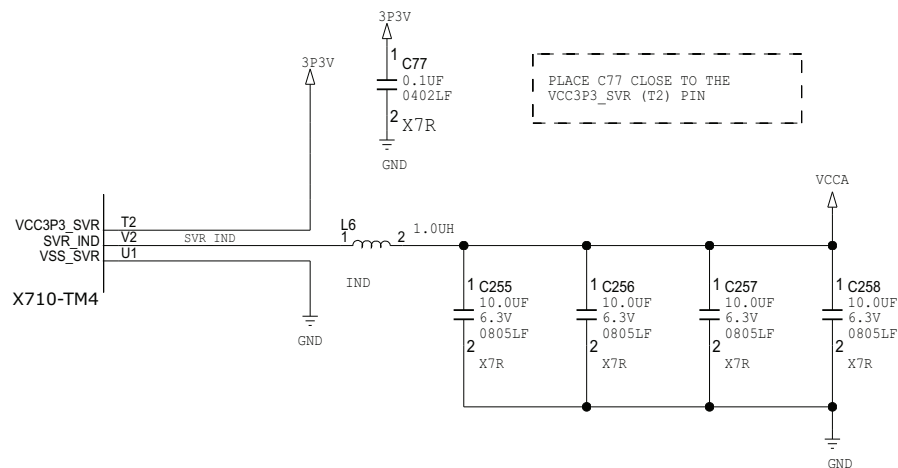


Figure 14-7. Connections to Implement VCCA On-die Regulation

14.12.1.1 Inductor

The chosen inductor should meet the following requirements.

- $L = 1 \mu\text{H} \pm 20\%$
- Irated > 1.6 A
- Isat > 2.2 A
- Rdc < 15 M Ω

When choosing an inductor designers need to take into consideration the inductor package size. To achieve optimal performance, the inductor should be as close as possible to the BGA package; however, certain size limitations might arise due to the geometry of the used heat sink. For a suggested list of inductors refer to [Table 14-8](#).

**Table 14-8. Suggested Inductors**

Part Number	L [uH]	Irated [A]	Isat [A]	Rdc-typ [mΩ]	Rdc-max [mΩ]
XFL4020-102ME	1	4.5	8	10.8	11.9
IHLP-2525BD-ER1R0M-01	1	9	16	13.1	14.2
IHLM-2525CZ-ER1R0M-01	1	11	22	9	10
IHLP-2525CZ-ER1R0M-01	1	11	22	9	10
MLC1538-102ML	1	12.4	24.5	3.46	3.81
MSS1038-102NL	1	7.3	9		6
SER1052-102ML	1	12.5	16.5		4

14.12.1.2 Output Capacitor

For the on-die switching voltage regulator to operate correctly, the VCCA analog rail should have 40 μF bulk capacitance attached to it. The plane should also have a few 100 nF capacitors attached for high frequency decoupling. The total capacitance of both bulk and high frequency decoupling capacitors, including tolerance, should not exceed 50 μF . It is recommended to use four capacitors 10 μF X7R or two capacitors 22 μF X7R. The total ESR < 1 M Ω and total Z < 1.5 M Ω @ F=1 MHz.

14.12.1.3 Input Capacitor

The input of this switching voltage regulator requires 40-100 μF bulk capacitance. It is recommended to use minimum four caps 10 μF X7R or minimum two caps 22 μF X7R. The total ESR << 1 M Ω and total Z << 1.5 M Ω @ F=1 MHz.

When designing the power source of the 3.3V rail (pins: U21, R11, T34, W11 and L9) designers should take into consideration the fact that this rail is supplying the input of the VCCA switching voltage regulator (pin U1). To avoid noise issues due to the input voltage ripple caused by the switching voltage regulator, Intel recommends that designers provide proper isolation between pin T2 and the rest of the 3.3V power pins. The same principle needs to be applied to the GND pin of the integrated power supply U1. Refer to the Power Supply Layout Recommendations section for more details.

14.12.2 VCCD Digital Supply

The X710-TM4 preferably implements a fixed VCCD voltage solution at 0.92V. Optionally, a VCCD legacy design using voltage scaling can be used. For a VCCD legacy design to operate efficiently, the regulator needs to ensure a 2% or tighter regulation of the VCCD power rail.

14.12.2.1 SVR Controller Selection

Given the high output current, the fixed SVR and VCCD Legacy variable voltage designs should compensate for the losses presented by the power delivery network. Ideally, this should be done by choosing regulators that offer differential remote voltage sensing providing feedback directly from the load. Furthermore, to prevent potential stability issues it is also recommended to use a current mode controller. The LTC3833 and the TPS40180 are good candidates.

Given that the regulator needs to provide an output voltage with a tighter than 2% accuracy, the chosen controller should have a 1% accurate (or better) reference voltage and should be low enough to enable adjusting the output voltage down to 0.75V.

14.12.2.2 SVR Implementation Example

This section describes a possible SVR implementation designed around the LTC3833 controller (see Figure 14-8).

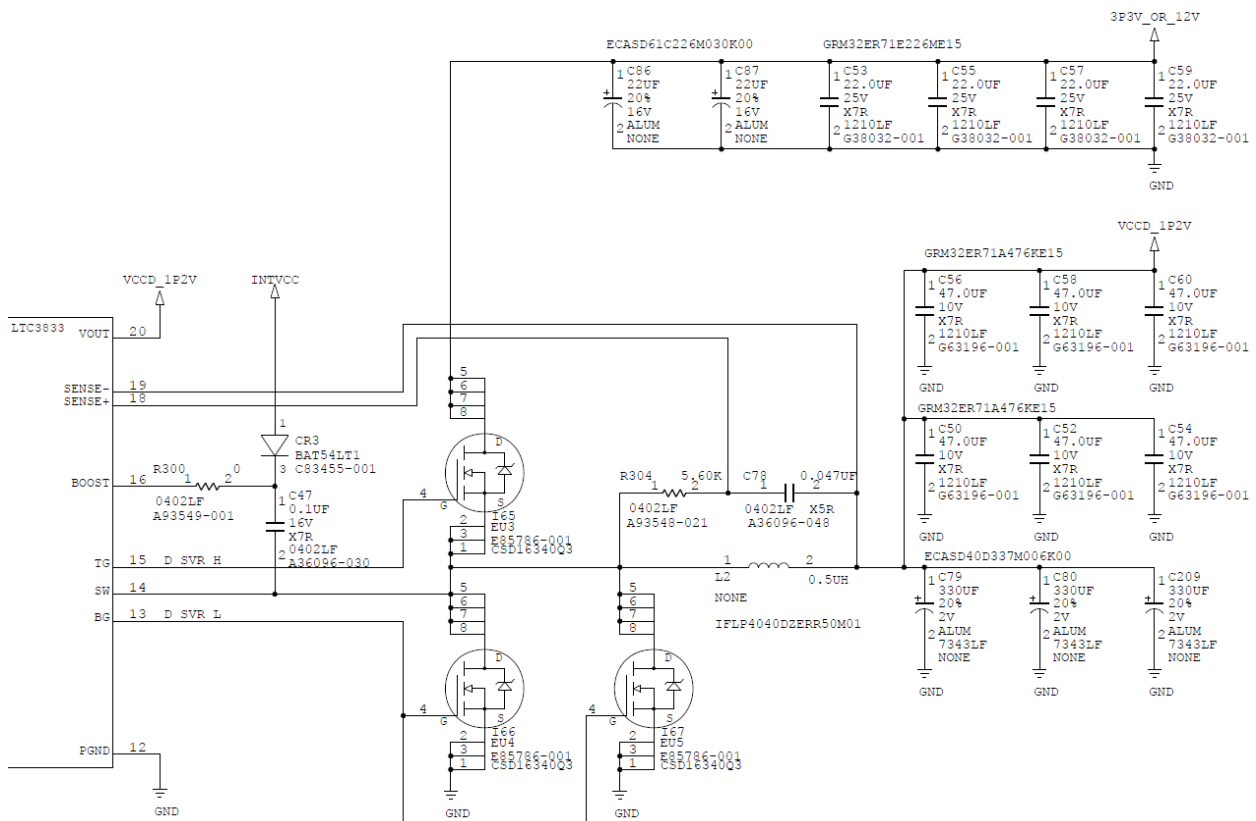


Figure 14-8. LTC3833 SVR Implementation Example



Given that the currently available the X710-TM4 power numbers are only estimates, the power stage of the regulator should be over dimensioned to ensure plenty of margin. Therefore, Intel recommends a regulator design capable of supplying a nominal load of 10 A and peak load of 12 A.

The operating frequency used is 300 KHz to avoid excessive switching losses.

Allowing for about 25% ripple current, the inductor value should be 0.5 μ H. It is critical to use an inductor with a low enough DC resistance to avoid over heating. The chosen inductor is the Vishay* IFLP-4040DZERR50M01 with a DC resistance of 0.88 m Ω . If an alternate part is chosen, the core and Rdc losses need to be re-evaluated to ensure thermal stability of the design.

For the top side switching element, the design is using Vishay SIS426DN and for the bottom side switching element the design is relying on 2x Vishay SIS426DN. The TI CSD16340Q3 FETs could be used as alternate parts.

To keep the input voltage ripple under control, the design is relying on a mixture of four Murata* GRM32ER71E226ME15 ceramic capacitors, and two Murata ECASD61C226M030K00 aluminum-polymer capacitors.

To ensure the output voltage ripple is less than 10 mV, the output capacitor tank was chosen to be a mixture of six Murata GRM32ER71A476KE15 ceramic capacitors, and three Murata ECASD40D337M006K00 aluminum-polymer capacitors.

The LTC3833 provides a differential voltage sensing amplifier that should be connected to the EXT_SVR_SENSE_P and EXT_SVR_SENSE_N pins through the previously mentioned feedback circuit. To set the output voltage correctly for the X710-TM4, the value of the resistors are as follows: $R_1=3K$, $R_2=12K$, and $R_4=91K$ (where R_4 is connected to the INTVCC rail of the LTC3833 controller).

For more information and a complete schematic, refer to the X710-TM4 reference schematics.

14.12.3 Power Supply Filtering

Provide several 1 μ F high-frequency bypass capacitors for each power rail. If possible, place the capacitors close to the load, possibly on the back side of the board directly underneath the BGA package and adjacent to power pads. For a list of recommended mix of bypass capacitors, refer to Table 14-9.

Traces between decoupling and I/O filter capacitors should be as short and wide as practical. Long and thin traces are more inductive and reduces the intended effect of decoupling capacitors. Vias to the decoupling capacitors should be sufficiently large in diameter to decrease series inductance. Alternatively, multiple vias connected in parallel can be used to connect the bulk capacitors to the different power planes.

Table 14-9. Minimum Number of Bypass Capacitors per Power Rail

Power Rail	Bulk Capacitance		High Frequency Bypass
	Cmin [μ F]	Cmax [μ F]	1 μ F
3.3V	84		8
VCCD	132		50
VCCA	40	50	32



14.12.4 Power Management and Wake Up

In order for the X710-TM4 to detect what type of power is available, designers must connect the MAIN_PWR_OK and the AUX_PWR signals on the board. These digital inputs are located in ball locations, E3 and E1, and serve the following purposes:

- MAIN_PWR_OK signals the X710-TM4 that the main power from the system is up and stable. For example, it could be pulled up to the 3.3V main rail or connected to a power well signal available in the system.
- When sampled high at power on reset, AUX_PWR indicates that auxiliary power is available to the X710-TM4, and therefore it advertises D3cold wake up support. The amount of power required for the function, which includes the entire network interface card, is advertised in the Power Management Data register, which is loaded from the NVM.

If wake-up support is desired, AUX_PWR needs to be pulled high and the appropriate wake-up LAN address filters must also be set. The initial power management settings are specified by NVM bits. When a wake-up event occurs, the X710-TM4 asserts the PE_WAKE_N signal to wake the system up. This signal remains asserted until PME status is cleared in the Power Management Control/Status Register.

14.13 Recommended Simulations for High Speed Serial Interconnects

KR and SFI signaling frequencies extend above 5 GHz. Relatively short stubs, small discontinuities, and fairly small in-pair trace length differences can cause signal integrity issues and an undesirable increase in bit errors. Before ordering circuit boards, verify that:

- Planned KR signal trace routing on the circuit board complies with the guidance provided in this document and the interconnect characteristics recommended in IEEE Std. 802.3ap sections 69.3 and 69.4.
- Planned SFI signal trace routing on the circuit board complies with the guidance provided in this document and complies with the interconnect characteristics recommended in the SFF-8431 specifications. Contact your Intel sales representative to get signal integrity support.
- Made use of Intel's available impedance calculators and that results match within 5% of calculated impedance from other tools. Contact your Intel sales representative to get signal integrity support.
- For most KR board traces, if possible, export S-parameters for the planned signal channels, and compare them to the IEEE recommended electrical characteristics. Optimize the signal path until it complies with the IEEE recommendations. IEEE channel characteristics recommendations are for the entire length of the board channels - from the solder-pads for one IC device to the solder-pads for another device, at the far-end of the entire channel path. This end-to-end signal path typically includes two or three circuit boards, connectors, and AC coupling capacitors.



- For unusual routing requirements, which make it difficult to meet the IEEE channel recommendations might still work satisfactorily with the X710-TM4. These are channels that have been optimized by following the layout guidelines recommended within this document but which cannot be improved enough to comply with IEEE Std. 802.3-2018 recommended electrical characteristics. With sufficient notice, Intel engineers can provide assistance. Trace routing should be optimized prior to following steps:
- Request a layout review from your Intel sales representative (must be willing to provide board stack-up information and trace CAD artwork).
- After traces have been optimized, if the IEEE recommended electrical characteristics are still not being met, then end-to-end board channel S-parameter models should be extracted (preferably in the Touchstone* S4p format) for additional investigative simulations by Intel signal integrity engineers. Please request the required S-parameter frequency range, step size etc., before extracting Touchstone S-parameter models.
- For accurate simulation results, pre and post plating information (including surface roughness) should also be included in the extracted channel models.

14.14 General Routing Guidelines

Intel has a layout guidance checklists available for the X710-TM4. Contact your local Intel representative for access.

14.14.1 Board Stackup

PCBs for designs typically have six, eight, or more layers. Although the X710-TM4 does not dictate stackup, the following examples are of typical stack-up options.

1. Microstrip example:

- Layer 1 is a signal layer.
- Layer 2 is a ground layer.
- Layer 3 is used for power planes.
- Layer 4 is a signal layer. Careful routing is necessary to prevent crosstalk with layer 5.
- Layer 5 is a signal layer. Careful routing is necessary to prevent crosstalk with layer 4.
- Layer 6 is used for power planes.
- Layer 7 is a signal ground layer.
- Layer 8 is a signal layer.

Note: Layers 4 and 5 should be used mostly for low-speed signals because they are referenced to potentially noisy power planes that might also be slotted.

2. Stripline example:

- Layer 1 is a signal layer.
- Layer 2 is a ground layer.
- Layer 3 is a signal layer.
- Layer 4 is used for power planes
- Layer 5 is used for power planes

- Layer 6 is a signal layer.
- Layer 7 is a signal ground layer.
- Layer 8 is a signal layer.

Note: To avoid the effect of the potentially noisy power planes on the high-speed signals, use offset stripline topology. The dielectric distance between the power plane and signal layer should be three times the distance between ground and signal layer.

This board stack-up configuration can be adjusted to conform to your company's design rules.

14.14.2 Power Supply

As previously mentioned in [Section 14.12](#), the 1.2V power plane needs to be split in two: a plane supplying the analog domain and one supplying the digital core.

14.14.2.1 VCCA Analog Supply

The analog domain's switching voltage regulator integrated in the X710-TM4 also needs attention. This switching voltage regulator is being powered from the 3.3V rail that serves as the power source for other noise sensitive blocks. The layout needs to provide separation between the noise due to the input voltage ripple and the rest of the 3.3V pins. [Figure 14-9](#) shows a VCCA layout example.

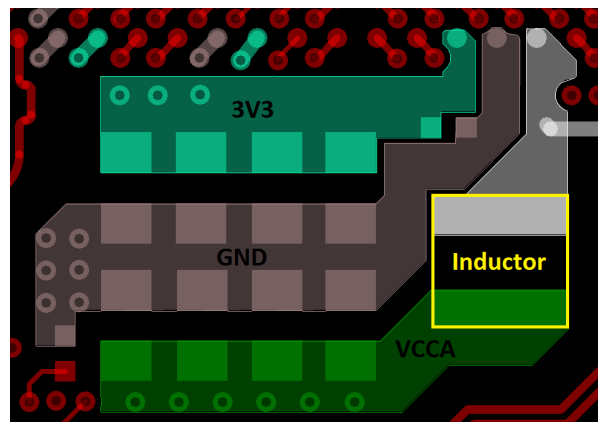


Figure 14-9. VCCA Layout Example

Note the bulk capacitance at the input of the regulator is placed close to the input pin and the rest of the 3.3V circuit is isolated from this copper island by a single contact point.

The same strategy can be used to avoid switching noise from propagating through the ground planes.

14.14.2.2 VCCD Digital Supply

The layout of the digital domain's power supply is also critical to ensure reliable operation.



The placement should focus on minimizing the high current loops in the power stage of the regulator and separating the feedback and compensation circuits from the noisy switching circuit. It is good to provide some separation between the X710-TM4 and the inductor.

The current sense signals should be routed differentially and separated from noisy planes like the switching node to avoid potential noise coupling into the signals and causing stability issues. The RC elements of this circuit need to be placed in the close vicinity of the X710-TM4 (within 300 mils).

Similarly, the voltage feedback signals should be routed differentially and separated from noisy nodes. The resistive feedback divider should be placed in the close vicinity of the X710-TM4 (within 300 mils).

The gate signals should be routed with wide traces and kept far away from the sensitive analog circuits.

The high current path should be routed with wide copper fills. Where layer transitions are necessary an array of vias should be used. The number of adequate vias is a function of the via-size and the peak currents but care should be taken to minimize the swiss-cheese effect of these vias on the affected power planes.

For more details, refer to the appropriate power supply controller's datasheet and design guidelines.

14.14.3 Miscellaneous

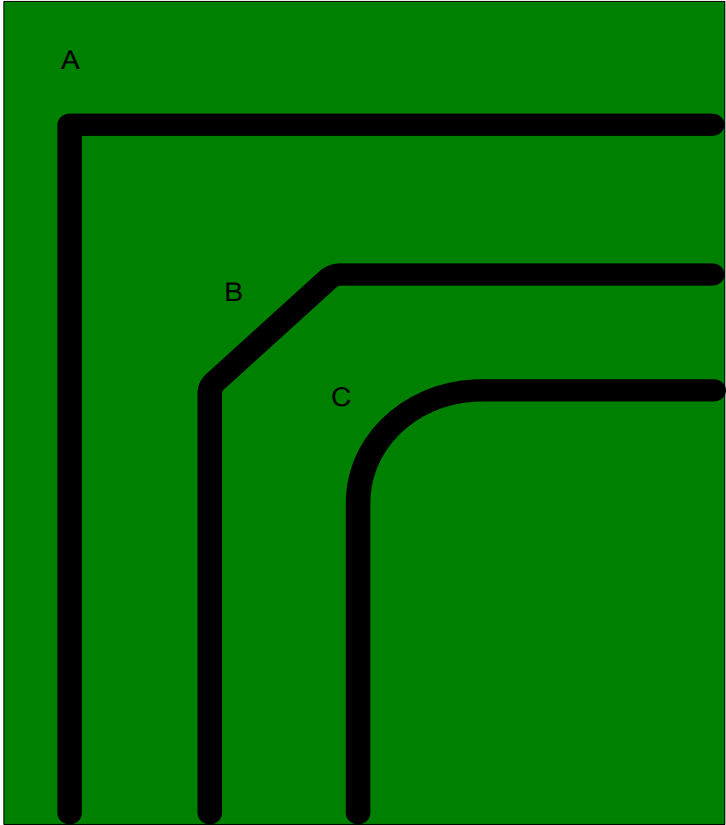
<p>Guideline</p>	<p>Following are three common options for routing 90 degree corners:</p> <ol style="list-style-type: none"> 1. Right angle routing with no change in trace width. This adds a small excess capacitance to the trace. It is a marginal technique and might be required in some instances such as in pin fields but options (b) or (c) are preferred for best signal integrity. 2. A slight miter or chamfer is added to reduce or eliminate the excess capacitance. This is automatically performed by most routers with simple option settings. 3. Manually routing to transact the orthogonal traces at 45° eliminates the excess capacitance and also minimizes trace length. Moreover, using round corners even further eliminates capacitance and provides shorter length. 
<p>Applicable Buses</p>	<p>All.</p>
<p>Purpose</p>	<p>Reduces length mismatch and discontinuity. Also reduces possibility of acid trap.</p>
<p>Significance</p>	<p>Medium. Large numbers of bends as shown in option (a) could cause accumulated common mode noise effects. Option (b) is good for most cases and option (c) is preferred for best signal integrity performance.</p>

Table 14-10. Pad Geometries

<p>Guideline</p>	<p>The following geometry is for reference. Follow the DFM rule if there is any conflict.</p> <ul style="list-style-type: none"> 0603 capacitor spacing. <ul style="list-style-type: none"> 30 x 35 mils PAD. 25 mils spacing (S) – can be up to 30 mils if test pad is required. 0402 capacitor. <ul style="list-style-type: none"> 20 x 20 mils PAD. 20 mils spacing (S) – can be up to 30 mils if test-pad is required. <p>For high speed interfaces, it is desired to use 0402 for smaller parasitics.</p> <div data-bbox="695 632 1162 995"> </div> <div data-bbox="699 1058 1162 1188"> </div> <div data-bbox="753 1251 1101 1486"> </div> <p>High speed connector solder pads should either be rounded or chamfered at the corners to eliminate sharp corners, which can cause reflections and concentration of high frequency currents at the corners.</p>
<p>Applicable Buses</p>	<p>All.</p>
<p>Purpose</p>	<p>Provides the reference design for capacitor pad and reduces reflections from sharp corners.</p>
<p>Significance</p>	<p>Medium.</p>
<p>Implementation</p>	<p>Follow the DFM rule if there is any conflict.</p>

14.15 Interface Specific Layout Considerations

14.15.1 NC-SI Layout Requirements

14.15.1.1 Board Impedance

The NC-SI signaling interface is a single ended signaling environment and as such Intel recommends a target board and trace impedance of 50 Ω +L20% and -10%. This impedance ensures optimal signal integrity and quality.

14.15.1.2 Trace Length Restrictions

The recommended maximum trace lengths for each circuit board application is dependent on the number drops and the total capacitive loading from all the trace segments on each NC SI signal net. The number vias must also be considered. Circuit board material variations and trace etch process variations affect the trace impedance and trace capacitance. For each fixed design, highest trace capacitance occurs when trace impedance is lowest. For the FR4 board stack-up provided in direct connect applications, the maximum length for a 50 Ω NC-SI trace would be approximately 9 inches on a -10% board impedance skew. This ensures that signal integrity and quality are preserved and enables the design to comply with NC-SI electrical requirements.

For special applications that require longer NC-SI traces, the total functional NC-SI trace length can be extended with non-compliant rise time by:

- Providing good clock and signal alignment
- Testing with the target receiver to verify it meets setup and hold requirements

For multi-drop applications, the total capacitance and the extra resistive loading affect the rise time. A multi-drop of two devices limits the total length to 8 inches. A multi-drop of four limits the total length to 6.5 inches. Capacitive loading of extra vias have a nominal effect on the total load.

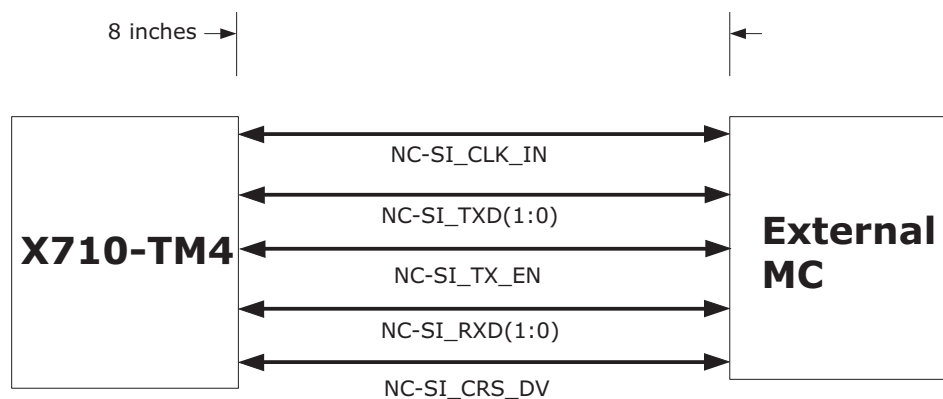


Figure 14-10. NC-SI Trace Length Requirement for Direct Connect



Table 14-11 lists how seven more vias increase the rise time by 0.5 ns. Again, longer trace lengths can be achieved.

Table 14-11. Stack Up (Seven Vias)

Item	Value	Units
Trace width	4.5	mils
Trace thickness	1.9	mils
Dielectric thickness	3.0	mils
Dielectric constant	4.1	--
Loss tangent	0.024	--
Nominal impedance	50	W
Trace capacitance	1.39	pf/inch

Table 14-12 lists the example trace lengths for the multi-drop topology of 2 and 4 shown in Figure 14-11 and Figure 14-12.

Table 14-12. Example Trace Lengths for Multi-Drop Topologies (2 and 4)

Multi-drop Length Parameter Used in Figure 14-11 and Figure 14-12	Segment Length Example Eor Multi-drop Configurations			
	2-drop Configuration		4-drop Configuration	
	Length (Inches)	Trace capacitance (Pf)	Length (Inches)	Trace capacitance (Pf)
L1	2	2.8	1.5	8.8
L2	4	5.6	2	16.1
L3	2	2.8	1	8.1
Total	8	11.1	6.5	35.6

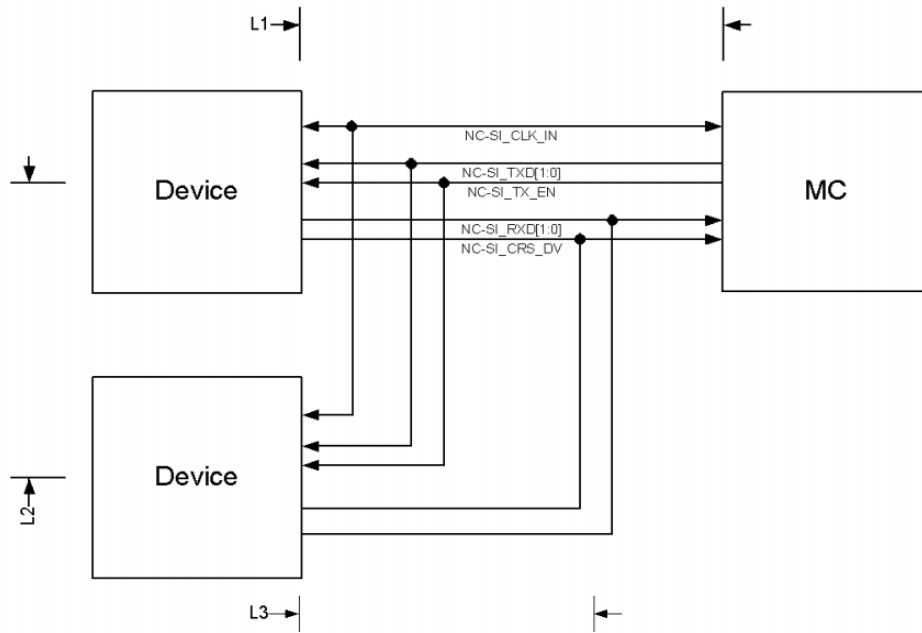


Figure 14-11. 2-drop Topology Example

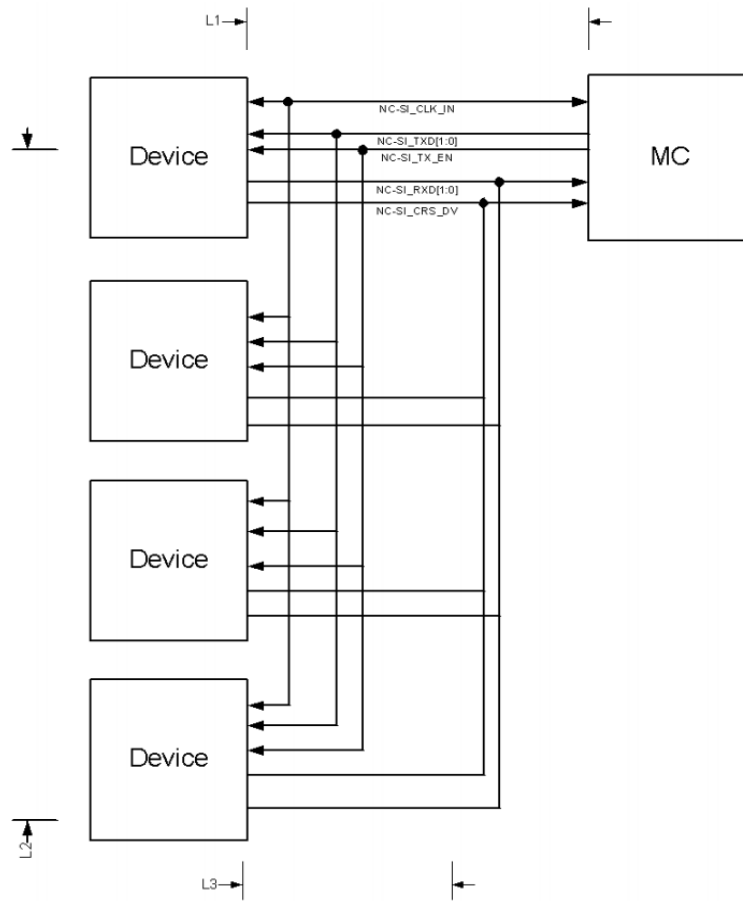


Figure 14-12. 4-drop Topology Example



Table 14-13. Compliant NC-SI Maximum Length on a 50 Ω -10% Skew-board with Example Stack-up

Topology	Total Maximum Compliant Linear Bus Size (Inches)	Number of Vias	Approximate Net Trace Capacitance Minus Load Capacitance (pf)
4 multi-drop	6.0	1	8.3
4 multi-drop	5.5	8	8.3
2 multi-drop	8.0	1	11.1
2 multi-drop	7.5	8	11.1
Point to point	9.0	1	12.5
Point to point	8.5	8	12.5

Extending NC-SI to a maximum 11 ns rise time increases the maximum trace length.

Table 14-14. Functional NC-SI Maximum Length on a 50 Ω -10% Skew Board with Example Stack-up (Based on Actual Lab-measured Solution)

Topology	Total Maximum Functional Linear Bus Size (Inches)	Number of Vias	Approximate Net Trace Capacitance Minus Load Capacitance (pf)
4 multi-drop	19	1	26.4
4 multi-drop	18	8	26.4
2 multi-drop	20	1	27.8
2 multi-drop	19	8	27.8
Point to point	22	1	30.6
Point to point	21	8	30.6

14.15.2 SFP+ Layout Recommendations

To meet strict TWDPc and DDJ electrical requirements, designers need to optimize SFI trace routing accordingly.

In order to determine the maximum trace lengths allowed for SFP+ in a design refer to the Intel® Ethernet Controller X710-TM4 SFP+ Differential Trace Calculator and the Intel® Ethernet Controller X710-TM4 SFI/SFP+ TX HSPICE Channel Checker Simulation Kit. [Table 14-15](#) lists some microstrip example trace geometries and lengths the X710-TM4 supports.

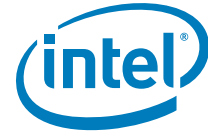


Table 14-15. SFP+ Trace Geometries

Trace Type	Dielectric Material	Dielectric Constant (dk or Er) 2.5 GHz to 5 GHz	Dissipation Factor (df or Loss Tangent)	Dielectric Layer Thickness (or Height) (mm)	Copper Trace Thickness After Plating ¹ (mm)	Max SFI Tx/Rx Trace Length (mm)	Main SFI Routing Trace Width (mm)	Main SFI Routing In-Pair Trace Separation, edge to edge (mm)	SFI Breakout Routing Trace Width ² (mm)	SFI Breakout Routing In-Pair Trace Separation, Edge-to-Edge (mm) ²
Microstrip	Nelco N4000-13 (2-ply 2113)	3.7	0.009	0.1905 (7.0 mils)	0.0508 (2.0 mils)	296.57 (11.676 in)	0.2921 (11.5 mils)	0.4064 (16 mils)	0.1778 (7 mils)	0.127 (5.0 mils)
Microstrip	Nelco N4000-13 (2-ply 2113)	3.86	0.009	0.1524 (6.0 mils)	0.0508 (2.0 mils)	269.69 (10.618 in)	0.2286 (9.5 mils)	0.3683 (14.5 mils)	0.1880 ³ (7.4 mils)	0.1905 (7.5 mils)
Microstrip	Panasonic Megtron6 (2-ply 3113)	3.63	0.003	0.2032 (8.0 mils)	0.0508 (2.0 mils)	461.39 (18.165 in)	0.3302 (13 mils)	0.4064 (16 mils)	0.2159 (8.5 mils)	0.1524 (6 mils)
Microstrip	Panasonic Megtron6 (2-ply 1080)	3.4	0.003	0.1524 (6.0 mils)	0.0508 (2.0 mils)	372.71 (14.674 in)	0.2667 (10.5 mils)	0.4572 (18 mils)	0.2032 (8 mils)	0.1905 (7.5 mils)
Microstrip	Isola FR408	3.63	0.013	0.1534 (6.04 mils)	0.0508 (2.0 mils)	233.12 (9.178 in)	0.2540 (10 mils)	0.4318 (17 mils)	0.1905 (7.5 mils)	0.1854 (7.3 mils)
Microstrip	Isola FR406	3.76	0.0186	0.1839 (7.24 mils)	0.0508 (2.0 mils)	200.07 (7.877 in)	0.2921 (11.5 mils)	0.4064 (16.0 mils)	0.1778 (7 mils)	0.1448 (5.7 mils)
Microstrip	Isola FR406	3.76	0.0186	0.1636 (6.44 mils)	0.0508 (2.0 mils)	188.21 (7.410 in)	0.2540 (10.2 mils)	0.3683 (14.5 mils)	0.1905 (7.5 mils)	0.1803 (7.1 mils)
Microstrip	FR4 2116, 2-ply	4.24 ³	0.024 ⁴	0.2286 ⁴ (9.0 mils)	0.0508 (2.0 mils)	171.34 (6.746 in)	0.3175 (12.5 mils)	0.3759 (14.8 mils)	0.1778 (7.0 mils)	0.1422 (5.6 mils)
Microstrip	FR4 2116, 2-ply	4.24 ⁴	0.024 ⁴	0.2286 ⁴ (9.0 mils)	0.04572 (1.8 mils)	171.39 (6.748 in)	0.3200 (12.6 mils)	0.3683 (14.5 mils)	0.1778 (7.0 mils)	0.1372 (5.4 mils)
Microstrip	FR4 2113, 2-ply	4.0	0.021	0.2032 (8.0 mils)	0.04826 (1.9 mils)	53.416 (7.202 in)	0.292 (11.5 mils)	0.356 (14.0 mils)	0.1674 (6.6 mils)	0.1371 (5.4 mils)
Microstrip	FR4 2113, 2-ply	4.05	0.021	0.2032 (8.0 mils)	0.04572 (1.8 mils)	56.718 (7.424 in)	0.305 (12.2 mils)	0.419 (16.5 mils)	0.1701 (6.7 mils)	0.1397 (5.5 mils)
Microstrip	FR4 2113, 2 Ply	4.0	0.021	0.1905 (7.5 mils)	0.04826 (1.9 mils)	53.416 (6.905 in)	0.2667 (10.5 mils)	0.2921 (11.5 mils)	0.2032 (8.0 mils)	0.1524 (6.0 mils)

1. Post-plating copper thickness tolerance ±0.3 mils (0.00762 mm).
2. For SFI traces that are 9 mils wide or less with 12 mils separation or less: Narrow breakout trace widths with smaller in-pair trace separation distances are discouraged. **Narrow SFI traces and/or less in-pair separation should NOT be required when the main trace route is 9 mils wide or less. This is especially true for the SFI Tx trace routes.**
3. 2116 FR4 manufactured by different vendors might have different dielectric constants, dissipation factors, different dielectric thicknesses (height), or a combination of these. Check with the appropriate circuit board supplier to obtain the correct properties (at 5 GHz) for the 2116 FR4 that is planned for use. If the 2116 FR4 has different electrical properties or different nominal thickness, the trace widths and trace separation might need to be adjusted.

For the breakout region of the X710-TM4, reducing the differential trace widths is allowed only if the main of the SFI routing trace width used is greater than 9 mils wide AND 100 Ω differential impedance is maintained in the breakout. If the main SFI trace routing uses 9 mils or smaller trace widths, then reducing SFI trace widths in the breakout region is not encouraged or even necessary. Table 14-15 lists recommended breakout region SFI trace and space dimensions for several different dielectric materials. In addition, when implementing narrow breakouts, keep the length of the breakout as short as possible (limiting the length of this narrow section to 100 mils for the SFI Tx pairs and 180 mils for Rx pairs). Figure 14-13 shows two narrow breakout examples and Figure 14-14 shows a Tx/Rx breakout example with 9 mil trace widths. If narrow or a thinner trace breakout is used, keep the narrow traces in-pair skew as low as possible using length matching techniques, this section is more susceptible to signal skew effects than the main SFI route.

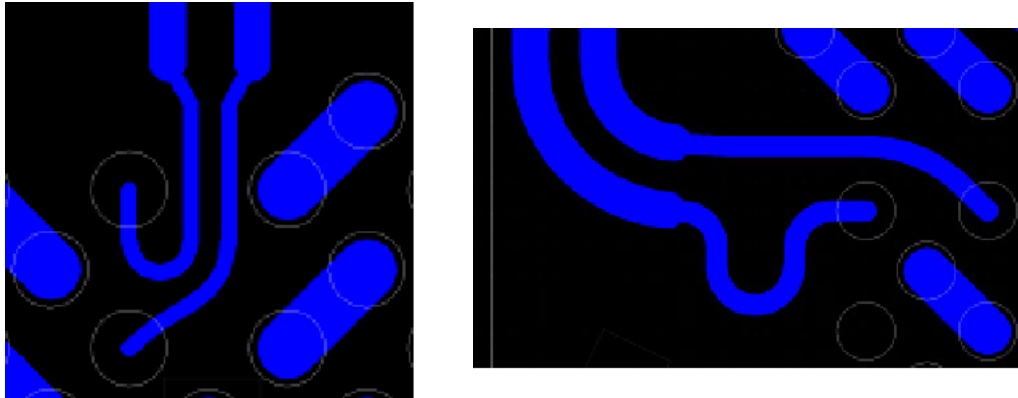


Figure 14-13. Narrow Breakout Routing

Note: Figure 14-13 shows thinner than recommended differential traces used for breakout. This routing is still allowed for this section. Note the breakout length matching.

Figure 14-13 left: Both traces are kept coupled with differential impedance around 100 Ω . This is only achievable in certain cases. Figure 14-13 right: Breakout traces are separated, allowing for wider traces, and each single trace's impedance is kept around 50 Ω .

Added trace loops for in-pair length matching should be located as close as possible to the X710-TM4's BGA solder pads. The space is very limited inside the breakout region so these loops can usually be made just outside the breakout section (see Figure 14-14).

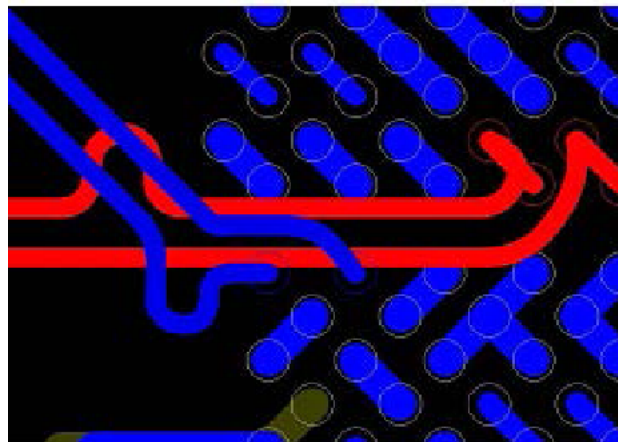


Figure 14-14. 9 Mil Wide Main Traces and Breakout

Note: There is no need for narrower breakout traces. Notice the added length matching trace loop is close to the breakout section in Figure 14-14. Red traces are on the bottom layer. The 82599 is on the top.



Intel strongly recommends against using serpentine traces to match the trace lengths within SFI signal pairs. With serpentes, it is much too easy to accidentally cause unintended signal skew and to increase the signal dispersion (undesirable pulse width and pulse edges spreading). Serpentes can be acceptable for PCIe in-pair trace length matching, if the serpentes are done correctly.

As shown in Figure 14-15, the trace lengths directly depend on the chosen lanes, placement of the X710-TM4 relative to the SFP+ cages and the placement of the SFP+ cages relative to each other. The placement can be defined by three variables as follows:

- Cage Gap—The distance between the cages
- BGA Offset X—Refers to the distance between the cages and the edge of the BGA package
- BGA Offset Y—Refers to the offset of the BGA package relative to the first cage

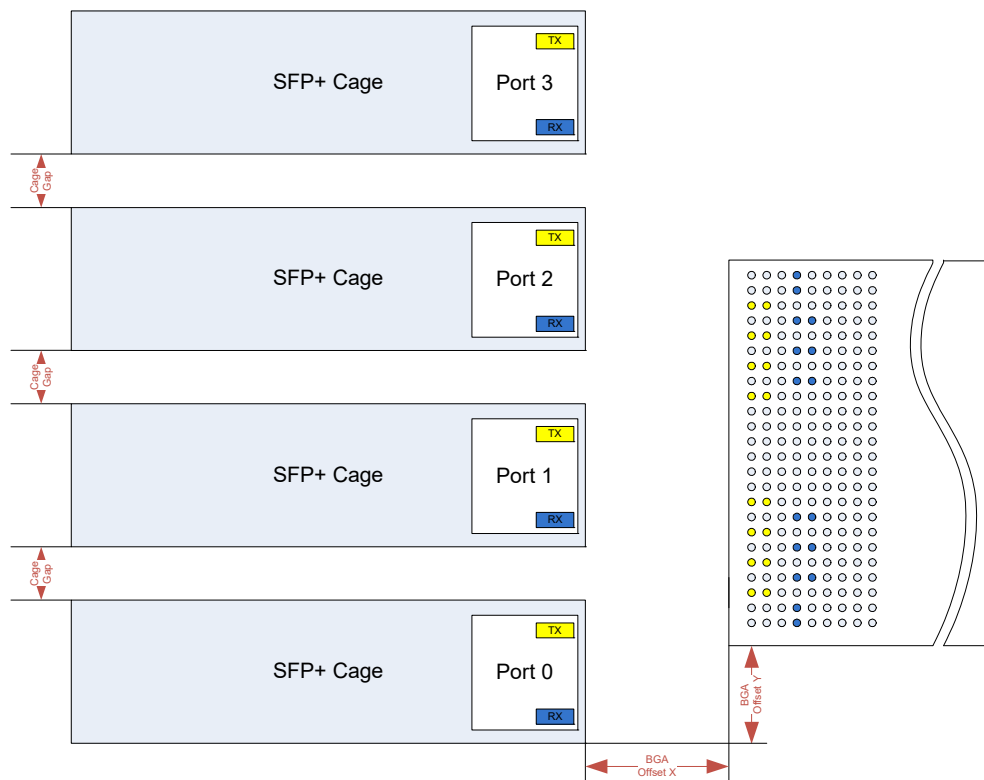


Figure 14-15. Component Placement

Table 14-16. SFP+ Connector Pad Voids

<p>Guideline</p>	<p>Make the voided areas under the connector pads rectangular-like in shape (chamfer or round the corners), and a little larger than the area occupied by each pair of SFP module connector pads. Keep the void width to 20 mils larger than the area occupied by the connector's signal pin solder-pads. The void dimensions should be approximately 110 mils long by 79 mils wide.</p> <ul style="list-style-type: none"> • Void similar-shaped areas of any inner Vcc power-plane that are under the signal pins or solder-pads of the SFI connector. The voids in power planes should be a few mils larger than the voids in the ground plane on the second layer. The reason for increasing the void dimensions on power planes layers is to prevent power supply noise from coupling onto the SFI signals. • Start plane voids ~10 mils before the SFI traces reach the connector solder-pad. The last 10 mils of the SFI traces should be over the reference plane void. If the void in the planes starts closer to the solder-pads, it causes fringe capacitance, which lowers the solder-pads' effective impedance too much. • Round or chamfer corners of the plane layer (this is especially important at the end of the voids where the SFI traces come in) to reduce possible reflections and EMI issues. If the plane void has sharp corners on the end where the SFI traces enter, it can cause high frequency currents to concentrate at those corners, and can potentially cause reflections and/or radiated EMI.
<p>Applicable Buses</p>	<p>SFI.</p>
<p>Purpose</p>	<p>Because the SFP+ module connector pads at the end of the SFI traces are wider than the SFI traces, the reference plane area directly under each pair of SFP module signal pin connector solder-pads must be voided in order to avoid excessive capacitance.</p>
<p>Significance</p>	<p>Low.</p>



14.15.3 KR Re-routing Layout Example

In some specific/custom use cases, there might be a desire to route the Ethernet interfaces to multiple locations on the board. Re-routing high speed serial signals requires careful design. One possible solution that a designer can explore is using AC coupling capacitors to implement the stuffing options. Placed carefully such that two capacitors share one pad that has a via in it, a designer can build a jumper structure that could enable re-routing of the high speed serial signals. See [Figure 14-16](#) for an example.

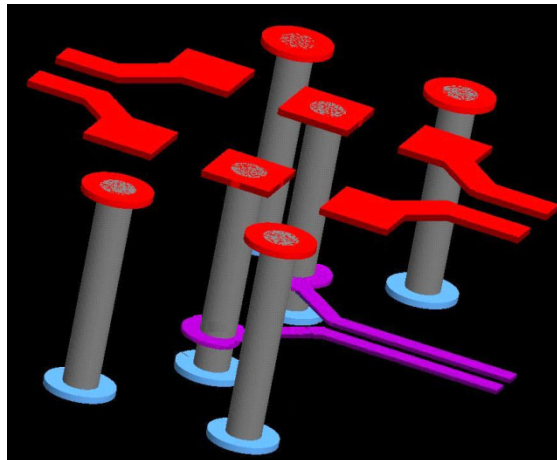


Figure 14-16. Routing High-speed Serial Signals

50 Ω single-ended impedance is necessary for transparent layer transitions from the signal's perspective. To achieve this, designers need to carefully place the return vias and might also need to void the reference plane around the signal vias and underneath the capacitor.

Please use 3D field-solver simulations to determine the appropriate structure for specific stack-ups.

14.16 Third-Party Magnetics Manufacturers

The following magnetics modules are recommended.

Component	Manufacturer	Part Number
Integrated Magnetic (1 x 1)	Bel Fuse*	G10-188T-012E
Integrated Magnetic (1 x 1)	Pulse*	JT11-1003HL
Integrated Magnetic (1 x 1)	Delta*	RLTC1K4360G-IN
Integrated Magnetic (1 x 4)	Pulse	JT14-1003HL



NOTE: *This page intentionally left blank.*



15.0 Thermal Design Considerations

15.1 Introduction

This can be used as an aid when designing a thermal solution for systems implementing the X710-TM4.

Properly designed solutions provide adequate cooling to maintain the X710-TM4 product case temperature T_{case} (or junction) at or below thermal specifications. The X710-TM4 should function properly if case temperatures are kept at or below those presented. Ideally this is accomplished by providing a low local ambient temperature airflow, and creating a minimal thermal resistance to that local ambient temperature.

By maintaining the case (or junction) temperature at or below the specified limits, a system designer can ensure the proper functionality, performance, and reliability of the X710-TM4. Operation outside the functional limits can cause data corruption or permanent damage to the X710-TM4.

The simplest and most cost-effective method to improve the inherent system cooling characteristics is through careful chassis design and placement of fans, vents, and ducts. When additional cooling is required, component thermal solutions can be implemented in conjunction with system thermal solutions. The size of the fan or heat sink can be varied to balance size and space constraints with acoustic noise.

15.2 Measuring Thermal Conditions

This section provides a method for determining the operating temperature of the X710-TM4 in a specific system based on case temperature. Case temperature is a function of the local ambient and internal temperatures of the X710-TM4. This document specifies a maximum allowable T_{case} for the X710-TM4.

15.3 Thermal Considerations

In a system environment, the temperature of a component is a function of both the system and component thermal characteristics. System-level thermal constraints consist of the local ambient temperature at the component, the airflow over the component and surrounding board, and the physical constraints at, above, and surrounding the component that may limit the size of a thermal enhancement (heat sink).

The component's case/die temperature depends on:

- Component power dissipation



- Size
- Packaging materials (effective thermal conductivity)
- Type of interconnection to the substrate and motherboard
- Presence of a thermal cooling solution
- Power density of the substrate, nearby components, and motherboard

All of these parameters are pushed by the continued trend of technology to increase performance levels (higher operating speeds, MHz) and power density (more transistors). As operating frequencies increase and packaging size decreases, the power density increases and the thermal cooling solution space and airflow become more constrained. The result is an increased emphasis on system design to ensure that thermal design requirements are met for each component in the system.

15.4 Importance of Thermal Management

The thermal management objective is to ensure that all system component temperatures are maintained within functional limits. The functional temperature limit is the range in which the electrical circuits are expected to meet specified performance requirements. Operation outside the functional limit can degrade system performance, cause logic errors, or cause device and/or system damage. Temperatures exceeding the maximum operating limits might result in irreversible changes in the device operating characteristics.

Note: The X710-TM4 doesn't provide autonomous on-die thermal management to monitor on-die temperature. The X710-TM4 doesn't react and doesn't shutdown when internal temperatures exceed specified thermal ratings. See [Table 13-1](#), [Table 13-2](#) and [Table 15-1](#) for more detail.

Also note that sustained operation at component maximum temperature limit might affect long-term device reliability.

15.5 Packaging Terminology

Item	Description
BLT	Bond Line Thickness. Final settled thickness of the thermal interface material (TIM) after installation of the heat sink.
CTE	Coefficient of Thermal Expansion. The relative rate a material expands during a thermal event.
FCmBGA	Molded Flip Chip Ball Grid Array package: A surface-mount package using a combination of flip chip and BGA structure whose PCB-interconnect method consists of Pb-free solder ball array on the interconnect side of the package. The die is flipped and connected to an organic build-up substrate with C4 bumps. The package is covered with mold to strengthen it and to protect the capacitors. The die is exposed for better thermal contact.
Junction	Refers to a P-N junction on the silicon. In this document, it is used as a temperature reference point (for example, θ_{JA} refers to the "junction" to ambient thermal resistance).
Ambient	Refers to local ambient temperature of the bulk air approaching the component. It can be measured by placing a thermocouple approximately 1"inch upstream from the component edge.



Item	Description
LFM	Linear Feet per Minute (airflow).
TA	Local ambient temperature
TJ	Junction temperature: maximum temperature of die active surface, i.e. hot spot.
TC	Case temperature: temperature at geometric center of dies or over mold top surface.
TDP	Thermal design power. The estimated maximum possible/expected power generated in a component by a realistic application. Thermal solutions should be designed to dissipate this power level. TDP is not the peak power that the component can dissipate.
TIM	Thermal Interface Material. A conductive material used between the component and heat sink to improve thermal conduction.
Θ_{JA} (Theta JA)	Thermal resistance junction-to-ambient, °C/W.
Ψ_{JC} (Psi JT)	Junction to case (top of package) thermal characteristic parameter, defined by $(T_J - T_C) / TDP$. Ψ_{JT} does not represent thermal resistance, but instead is a characteristic parameter that can be used to convert between T_J and T_{case} when knowing the total TDP. This parameter can vary by environment conditions like heat sink and airflow.

15.6 Thermal Specifications

To ensure proper operation of the X710-TM4, the thermal solution must maintain a case temperature at or below the 110 °C. System-level or component-level thermal enhancements are required to dissipate the generated heat to ensure the case temperature never exceeds that maximum temperature.

Good heat sink and good system airflow is critical to dissipate the X710-TM4’s high power. The size and number of fans, vents, and/or ducts, and, their placement in relation to components and airflow channels within the system determine airflow. Good Thermal Interfaces Material (TIM) between the heat sink and die should be applied correctly.

To develop a reliable, cost-effective thermal solution, all of the system variables must be considered. Use system-level thermal characteristics and simulations to account for individual component thermal requirements.

Keep the following in mind when reviewing the data that is included in this datasheet:

- All data is preliminary and is not validated against physical samples.
- Your system design might be significantly different.
- A larger board with more layers might improve the X710-TM4 thermal performance.

Table 15-1. X710-TM4 Thermal Specifications

Parameter	Specification	Notes
T_{J-MAX}	110 °C	
T_{C-MAX}	109 °C	At center of die top.
T_{C-MIN}	0 °C	
Ψ_{JC}	0.1 °C/W	Junction to case (top of die) thermal characteristic parameter.
TDP	W	9.748



15.6.1 2R Model Parameters

Θ_{JB} (°C/W)	Θ_{JC} (°C/W)
6.582	0.688

Thermal Analysis	
Ambient Temperature (C):	25
Flow Velocity:	0, 1, and 2.5 m/s
Motherboard Type:	JEDEC 1S2P (Standard)
Motherboard Size:	101.6 mm x 101.6 mm
Motherboard Thermal Vias:	100
External Heatsink:	Medium

15.6.2 Still Air JEDEC Environment

Package Thermal Characteristics in Standard Still Air JEDEC Environment for Reference	
Package	Θ_{JA} (°C/W)
No HS	14.01
With HS 35 mm	9.73

Note: Θ_{JA} is the thermal resistance junction-to-ambient of the package.

The thermal parameters previously defined are based on simulated results of packages assembled on standard JEDEC boards in a natural convection environment (still air).

The X710-TM4 cannot operate properly without a heat sink or any other good cooling method.



15.6.3 Package Thermal Characteristics with Forced-air JEDEC Environment

The thermal parameters that follow are based on simulated results of packages assembled on standard multilayer 1s2p 1.0-oz Cu layer boards in a forced-air JEDEC chamber. A system with the following attributes was used to generate thermal characteristics data:

- Standard JEDEC forced air environment
- Extruded aluminum heat sink
 - 44 x 44 x 6 mm base
 - 144 fins Z = 35 mm
- TIM (standard thermal grease)
- 101.6 mm x 101.6 mm PCB

Thermal Analysis	
Ambient Temperature (C):	25
Flow Velocity:	0, 1, and 2.5 m/s
Motherboard Type:	JEDEC 1S2P (Standard)
Motherboard Size:	101.6 mm x 101.6 mm
Motherboard Thermal Vias:	100
External Heatsink:	Medium

Velocity (m/s)	Θ_{JA} (°C/W)	Tj 1 (°C)	Tj 2 (°C)
0	9.728	125.59	132.01
1	4.097	63.6	70.07
2.5	2.449	45.65	51.94

User system design might vary considerably from the typical system board environment used to generate these figures.

Note: Flotherm 2-resistor thermal model is available upon request.

Contact your local Intel sales representative for the X710-TM4 thermal model.



15.7 Package Mechanical Attributes

Refer to [Section 13.7](#) for mechanical package detail. [Section 13.7](#).



16.0 Glossary and Acronyms

This section defines terms and acronyms commonly used throughout this specification. Another source for this type of information can be found in [Section 1.4](#), which contains some of the acronyms defined in the industry standards. For example, MII, MPA, Verbs, NC-SI, EEE, etc).

Table 16-1. Glossary and Acronyms

Auto-negotiation (AN) – An IEEE Ethernet method for exchanging PHY layer parameters such as speed and duplex mode between link partners.
Base Address Register (BAR) – Standard registers defined in PCI Config space that define how an I/O adapter responds to host memory-mapped I/O requests
Bit Error Rate (BER) – Number of bits received in error, divided by the total number of bits received.
Baseboard Management Controller (BMC) – A BMC is a specialized embedded processor typically integrated on a server motherboard that reports the state of the server to the system administrator independent of the state of the server OS.
Completion (Completed, Completes, etc) – When an RNIC has performed all functions specified for a given WQ operation, including Placement and Delivery, that WQ operation is said to be “completed”. This can be determined by the Consumer through a Work Completion for Signaled Work Requests.
Completion Event Queue (CEQ) – The X710-TM4 writes Completion events to this shared queue to make it easy for software to determine which CQ has new CQEs.
Completion Queue (CQ) – A sharable queue which can contain one or more Completion Queue Entries. A Completion Queue is used to create a single point of completion notification for multiple Work Queues. The Work Queues associated with a Completion Queue may be from different QPs and of differing queue types (SQs or RQs).
Completion Queue Entry (CQE) – Info that the X710-TM4 writes onto a Completion Queue during the process of performing a Completion.
Consumer – A software process that communicates with the X710-TM4. The Consumer typically consists of an application program or an OS adaptation layer which provides some OS-specific API.
Control / Status Register (CSR) – I/O adapter registers typically accessed by the host using memory-mapped I/O requests.
Cyclic Redundancy Check (CRC) – An error detecting code commonly used to protect network transmissions.
Embedded Management Processor (EMP) – A the X710-TM4 embedded processor that handles device initialization and also device configuration based on commands received from an Admin Queue, a management interface (e.g. NC-SI), or from a network port.
Error Correcting Code (ECC) – A code commonly used to protect network transmissions which allows both detection and recovery from errors.
Function-level Reset (FLR) – A capability defined in the <i>PCI Express Base Specification</i> that enables software to quiesce and reset a particular PCI Function in a MFD, without affecting the other PCI functions therein.
Function Private Memory (FPM) – The X710-TM4 uses host memory as backing store for a number of context objects such as queue state and iWARP objects. These objects are cached in the X710-TM4 Host Memory Cache (HMC). Each X710-TM4 PF or VF driver allocates host pages for this backing store according to its needs. Host pages are mapped into a virtually contiguous Private Memory address space that the HMC uses for object access. A contiguous portion of Private Memory address space is allocated for each PCI Function, and is referred to as that PCI Function’s <i>Function Private Memory</i> (FPM). All of the host pages allocated by a given PF or VF driver are mapped into its FPM.
Inter Packet Gap (IPG) – a set of idle signals sent over the network to separate packets
Interrupt Throttling (ITR) – A method of interrupt moderation that guarantees a minimum gap between two consecutive interrupts.



Table 16-1. Glossary and Acronyms (Continued)

Keyboard, Video, Mouse (KVM) – Standard bundle of computer user I/O devices
LAN On Motherboard (LOM) – When the X710-TM4 is integrated on a server motherboard, it qualifies as a LOM NIC.
Link Aggregation Control Protocol (LACP) – The IEEE protocol that enables the formation of Link Aggregation Groups (aggregation of one or more Ethernet links into a single logical link).
Link Layer Discovery Protocol (LLDP) – The IEEE protocol that enables a server to advertise its identity, capabilities, and interconnections to other entities on an Ethernet fabric.
Lower Layer Protocol (LLP) – The protocol layer beneath the protocol layer currently being referenced. For example, for TCP the LLP is IP. Etc.
Management Data Input/Output Interface (MDIO) – A standard interface to connect a MAC to a PHY, used to access PHY registers.
Maximum Segment Size (MSS) – The largest amount of data that a network device can handle in a single, unfragmented piece.
Memory Window (MW) – A subset of a Memory Region, which can be remotely accessed in a logically contiguous fashion. A Memory Window is identified by an STag, a Base TO, and a length, but also references an underlying Memory Region and has Access Rights.
Multi-Function Device (MFD) – A PCI Device with more than one PCI function
Multi-Function per Port (MFP) – A PCI Device is classified as an <i>MFP device</i> when it can support more than one PCI Physical Function bound to a single network port. See also <i>Single-Function per Port (SFP)</i> .
Nearest Bridge Address - a multicast MAC address used for DCBx exchange.
Network Interface Controller (NIC) – An I/O adapter that enables a computer to communicate over a network.
Non-TPMR Address - non-two-Port MAC Relay - a multicast MAC address used for CDCP LLDP exchange.
Packet – A unit composed of headers, data and footers that are sent or received by a device. Also known as a frame.
Page List – A list of physical addresses describing a set of memory pages, which specifies the page size, list of physical addresses, and offset to the start of the memory region within the first page. The starting physical addresses of each page is aligned on power-of-two addresses and the size of the page is a power of two. Note that it is possible for the starting offset to be an offset into the first page and to be of a byte granularity and the entire list may have an arbitrary length.
Physical Buffer List (PBL) – In iWARP terminology, a Physical Buffer List can either be a Block List or a Page List. The X710-TM4 does not support Block Lists, so in the context of the X710-TM4, PBLs and Page Lists are the same thing.
Physical Function (PF) – A PCI Function that supports the PCI-SIG <i>Single Root I/O Virtualization and Sharing Specification</i> .
Quad – In the X710-TM4 HAS, the term “quad” is defined as the set containing {source IP address, dest IP address, source port, dest port} taken from a TCP/IP packet.
Scheduler Quanta - A “quanta” is a configurable amount of bytes which may consist of one or more packets. The Scheduler schedules a “quanta” in each scheduling decision.
Receive Side Scaling (RSS) – A feature of Microsoft Windows OS that distributes receive packet processing to the different processors in a multi-processor system. Received packets are classified by the X710-TM4 under OS control into groups of conversations. Each group of conversations is assigned its own receive queue and receiving processor.
Receive Queue (RQ) – One of the two Work Queues associated with a Queue Pair. The Receive Queue contains Work Queue Elements that describe the Buffers into which data from incoming Send Operation Types is placed.
RNIC – The generic term for a device that implements iWARP verbs functionality. The X710-TM4 is an RNIC.



Table 16-1. Glossary and Acronyms (Continued)

<p>Single-Function per Port (SFP) – A PCI Device is classified as an <i>SFP device</i> when it can support only one PCI Physical Function bound to a single network port. See also <i>Multi-Function per Port (MFP)</i>.</p>
<p>Small Form-factor Pluggable (SFP) – A <i>small form-factor pluggable (SFP or SFP+)</i> module is a compact, hot-pluggable transceiver that interfaces a network device like the X710-TM4 to a fiber optic or copper networking cable.</p>
<p>Software Definable Pins (SDPs) – The X710-TM4 device pins that have a high degree of software configurability and programmability. Also called “general purpose I/Os” (GPIOs).</p>
<p>Type, Length, Value (TLV) – A technique used to encode messages in a data communication protocol. Each message is comprised of three sequential fields: A <i>Type</i> field, fixed in size and typically ~1 byte, which identifies the specific type of message and the format of information in the <i>Value</i> field. A <i>Length</i> field, fixed in size and typically ~1 byte, which defines the length of the <i>Value</i> field in octets. A <i>Value</i> field, variable in size, which contains all of the data specific to this message type. A protocol relevant to the X710-TM4 that uses TLV encoding is IEEE Link Layer Discovery Protocol (LLDP).</p>
<p>Transmit Segmentation Offload (TSO) – Also known as Large Send Offload (LSO). This high performance NIC feature allows the host OS to pass large chunks of data payload to the NIC with instructions on how to segment the payload into multiple packets for transmission.</p>
<p>Upper Layer Protocol (ULP) – The protocol layer above the protocol layer currently being referenced. For example, for IP the ULP is TCP. Etc.</p>
<p>Virtual Ethernet Bridge (VEB) – This is an IEEE EVB term. A VEB is a VLAN Bridge internal to the X710-TM4 that bridges the traffic of multiple VSIs over an internal virtual network.</p>
<p>Virtual Ethernet Port Aggregator (VEPA) – This is an IEEE EVB term. A VEPA multiplexes the traffic of one or more VSIs onto a single the X710-TM4 Ethernet port. The biggest difference between a VEB and a VEPA is that a VEB can switch packets internally between VSIs, whereas a VEPA cannot.</p>
<p>Virtual Function (VF) – A VF is a PCI Function that supports the PCI-SIG <i>Single Root I/O Virtualization and Sharing Specification</i>. One or more VFs are associated with a single PF. The group of VFs/PFs shares one or more physical resources, such as an Ethernet port. A VF is designed to be programmed by a Virtual Machine, whereas a PF is designed to be programmed by a higher-privileged entity, such as a Hypervisor.</p>
<p>Virtual Machine Devices queues (VMDq) – Virtual Machine Devices queues (VMDq) is a collection of NIC features designed to offload the hypervisor from performing classification and distribution of received packets to the Virtual Machines under its control. There are two versions of VMDq: VMDq1 and VMDq2. VMDq2 is a superset of VMDq1. Its major new features are: internal switching from a Transmit Queue to a Receive Queue, the ability to replicate received multicast and broadcast packets to multiple Receive Queues, the ability to sort received packets based on a combination VLAN tag and MAC address filter, and anti-spoofing transmit filters.</p>
<p>Virtual Machine Monitor (VMM) – A software component that allocates/exports/isolates server resources to the Virtual Machines (aka System Images). Other terms for VMM in this specification: Hypervisor, Virtualization Intermediary (VI).</p>
<p>Vital Product Data (VPD) – VPD is defined in the <i>PCI Local Bus Specification</i> (this is the conventional PCI spec, not PCIe). VPD is information that uniquely identifies hardware and software elements of a server. The VPD provides a server OS with information on various Field Replaceable Units such as part number, serial number, etc.</p>
<p>Virtual Station Interface (VSI) – This is an IEEE EVB term that defines the properties of a virtual machine’s (or a physical machine’s) connection to the network. Each downstream v-port on a the X710-TM4 VEB or VEPA defines a VSI. A standards-based definition of VSI properties enables network management tools to perform virtual machine migration and associated network re-configuration in a vendor-neutral manner.</p>
<p>Wake on LAN (WoL) – Wake on LAN is an Ethernet technology that enables a computer to power on or “wake up” upon receipt of a network message, often called a “magic packet”. Newer specifications in this area call WoL <i>Advanced Power Management wake up</i>.</p>
<p>Weighted Round-Robin (WRR) – A scheduling or arbitration algorithm which selects amongst requesters in a round-robin fashion, and in which each requester can be programmed to receive a different percentage share of resource bandwidth.</p>



Table 16-1. Glossary and Acronyms (Continued)

Weighted Strict Priority (WSP) – A scheduling or arbitration algorithm which selects the requester with highest priority that has not exhausted its programmed percentage share of resource bandwidth.
Work Queue (WQ) – One of either a Send Queue or Receive Queue.
Work Queue Element (WQE) – A Work Request transformed by software into X710-TM4-specific format, and enqueued on a Work Queue.
Work Request (WR) – A request to the X710-TM4 by the Consumer to perform some operation. Gets transformed by software into a Work Queue Element.



Appendix A Transmit Scheduling

This section provides a high level description of internal scheduling structures, scheduling concepts and algorithms deployed by the X710-TM4 scheduler. Standard software device drivers do not have an ability to program the scheduler directly and they are limited to the programming interface exposed by the admin queue described in [Section 7.8.4](#).

This section enables a privileged software device driver developers take advantage of the full scope of scheduler capabilities and not be limited by standard configurations exposed by the programming interface.

This section can be useful for an advanced reader (firmware engineers) that are willing to have a better understanding of scheduler operation.

A.1 Hierarchical Scheduling Concept

A.1.1 Abstract Scheduling Tree

This section describes a concept of the tree scheduling, terminology and logical scheduling tree structure.

[Figure A-1](#) shows an abstract tree structure. The structure of this tree and terminology is described in the text that follows.

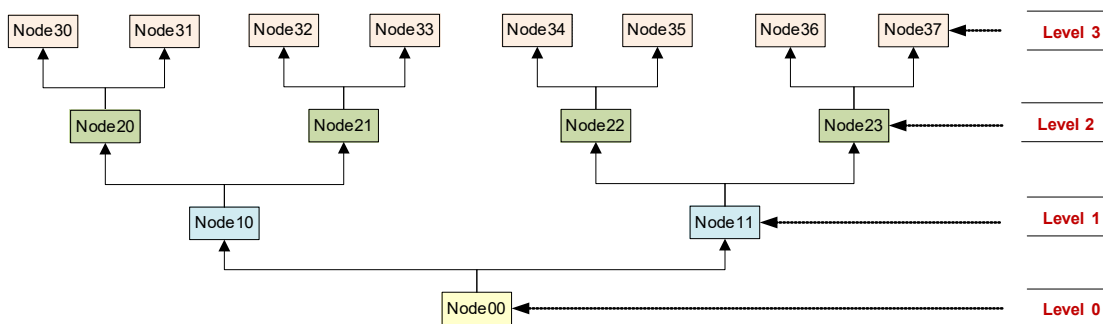


Figure A-1. Abstract Tree Structure

Tree components:

- Node. Node is a basic tree element. Each tree node might have at most one parent node and one or more child nodes. The definition of tree configuration concludes establishing a parent-child relation between all tree nodes.



1. Root node. A tree node that does not have a parent node. Root node occupies the lowest level of the tree (level 0). Node00 is a root node of the tree in the previous example. Scheduling tree traversals always starts from the root node.
 - Parent node. A tree node that has one or more child nodes. Parent node is always located on the lower tree Level with respect to child nodes. Node10 is a parent node, with two child nodes Node20 and Node21. Parent node leads to its children node at scheduling tree traversal.
 - Child node. A tree node that has a parent node. Note that child node might be a parent node with respect with other Nodes. Node20 is a child node to Node10, and parent node with respect to Node30.
 - Sibling nodes. Tree nodes that are child nodes of the same parent node. All sibling nodes are located on the same tree level. Nodes Node20 and Node21 are sibling nodes, having Node10 as a parent node. Scheduling tree traversal involves selection of one of the siblings in the tree to proceed traversal.
 - Leaf node. Tree node that does not have any child node. Leaf node is located on the highest level of the Tree. Node30 is a leaf node. Leaf node is an end point of scheduling tree traversal, and refers to the actual work that needs to be done.
 - Tree level. Tree nodes that are located on the same tree elevation. Nodes located on the same level not necessarily have same parent node, since they might belong to different sub-trees. Nodes Node20 and Node22 are located on the same tree level (level 2) but they belong to different subtrees, since they have different parent nodes, Node10 and Node11, respectively.
 - Subtree. Portion of the tree comprised of the tree node as a root node and its descendants. Subtree of Node10 includes Node20, Node21, Node30, Node31, Node32, Node33.
 - Height of the tree. Number of tree levels + 1. Height of the tree is a number of steps in the scheduling tree traversal.
 - Blocked node. A node is considered to be blocked if it cannot satisfy scheduling constrains applied, or all child nodes of that node are blocked. Leaf node can also be blocked because it does not have work available in any of transmit descriptor queues associated with it. Blocked node cannot be included in scheduling tree traversal. This blocked node propagation characteristic guarantees that if the node is not blocked, then there exists traversal from this node to one of the leaf nodes, where all nodes on that traversal are not blocked.

A.1.2 Tree-based Scheduling

Tree based scheduling is a process of finding a next leaf node to be scheduled. Effectively it involves finding a path in the tree leading from the root node to one of non-blocked leaf nodes, by traversing tree nodes from parent to child. A found path cannot consist of any blocked nodes. A blocked node cannot be included to the scheduling traversal. If parent node is not blocked, then one of child nodes should not be blocked ever. This guaranties that if a node has been selected to scheduling traversal, there is a path from that node leading to the leaf node and consisting of non-blocked nodes. A root node serves as a base of the tree. Leaf nodes are the only nodes that are referring to the schedulable work items. The rest of the tree nodes are used to enable flexible hierarchical arbitration and resource distribution.

Following is a definition of the scheduling tree traversal algorithm:

1. Select root node.
2. Go to the next tree level. For example children nodes.
3. Select one of the sibling nodes.



4. Selection of the sibling node depends on arbitration scheme and constrains applied on each of the siblings. See [Appendix A.1.3](#) and [Appendix A.1.5](#) for a description of the arbitration schemes and bandwidth distribution mechanisms.
5. If a selected node is a leaf node, scheduling tree traversal is completed.
6. Otherwise go to step 2.

Following is an example of the scheduling traversal in the tree shown in [Figure A-1](#).

- Select root node (Node00).
- Climb to the next tree level (level 1), and select a node among siblings (Node10 and Node11). For example, Node10.
- Climb to the next tree level (level 2), and select a node among siblings (Node20 and Node 21). For example, Node21.
- Climb to the next tree level (level 3), and select a node among siblings (Node32 and Node33). For example, Node33.
- Node33 is a leaf node and tree traversal is complete.

A.1.3 Bandwidth Allocation and Bandwidth Limit

The X710-TM4 supports two bandwidth distribution mechanisms that can be applied to any node in the scheduling tree. These mechanisms define how parent node resources are used and shared by the child nodes. Both constrains are stated in the terms of bandwidth. [Figure A-2](#) shows both concepts and their difference.

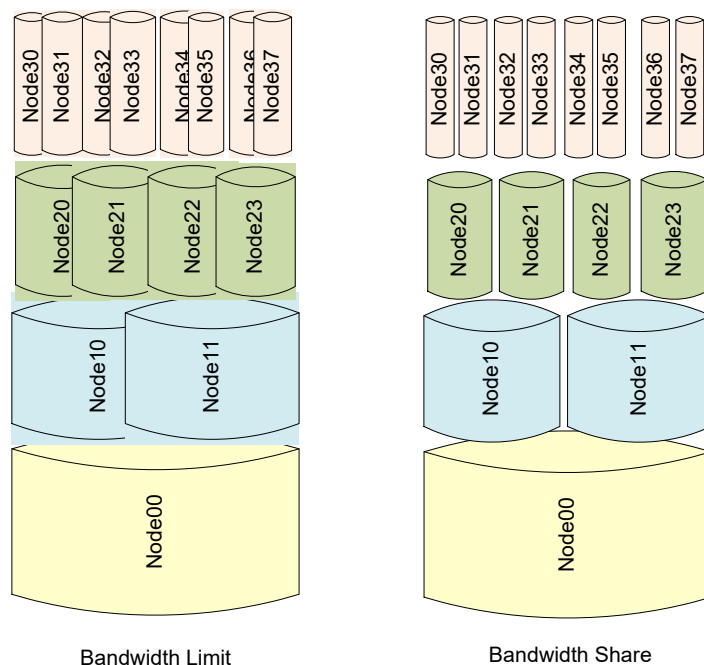


Figure A-2. Bandwidth Limit and Bandwidth Allocation



- Relative bandwidth allocation or bandwidth share. Allows to specify a relative share of available bandwidth among sibling nodes. Sibling nodes are sharing a total amount of bandwidth available from the parent node. The share of the parent bandwidth is relative, both with respect to the total amount of bandwidth available and use of the bandwidth by other sibling nodes. If one of the sibling nodes cannot use its bandwidth share, the unused bandwidth can be used by other sibling nodes. Once a node used its bandwidth share, it cannot be scheduled and considered to be blocked until all non-blocked sibling nodes had a chance to use their bandwidth share. Bandwidth share can be expressed in terms of weights or credits assigned to each tree node.
- Sibling nodes in the tree are sharing bandwidth of the parent node. For example, Node20 and Node21 are sharing bandwidth of Node10. If the share of the Node20 is 20% and share of Node21 is 80% and if parent Node10 allows 10 Gb/s, then the bandwidth available for the Node20 is 2 Gb/s, and bandwidth available for the Node21 is 8 Gb/s. If parent Node10 allows 5 Gb/s, then the bandwidth available for the Node20 is 1 Gb/s, and bandwidth share of the Node21 is 4 Gb/s.
- Bandwidth share can be individually assigned to each node in the tree.
- Best effort or zero-bandwidth allocation or bandwidth share. While relative bandwidth allocation previously described guarantees a relative part of the bandwidth to be always available for the sibling node, best effort or zero-bandwidth allocation does not guarantee any bandwidth allocation for the node, but allows this node to use a portion of the remaining bandwidth after all sibling nodes with relative bandwidth allocation had used their bandwidth share.
- For example, Node20 has assigned 50% of the Node10 bandwidth and Node21 has a zero-bandwidth allocation. In this configuration, assuming that both nodes have always data to transmit, Node20 uses its 50% of the Node10 bandwidth allocation first and then both nodes share the remaining 50%. As a result, Node20 is using 75% and Node21 is using 25% of Node10 bandwidth.
- Bandwidth limit. Allows to specify a maximum bandwidth that can be consumed by the tree node. A child node cannot consume more bandwidth than allowed by the bandwidth limit of the parent node. However sum of the bandwidth limits of the children nodes can exceed the bandwidth limit of the parent node and potentially allow better use of the available bandwidth. [Figure A-2](#) shows this concept, where child nodes represent an overlapping virtual pipes contained within parent node virtual pipe.
- Each tree node might have an individually assigned bandwidth limit. Some nodes in the tree might need to share the bandwidth limit.
- Shared bandwidth limit. Allows to specify a bandwidth limit for the group of nodes that are not co-located in the tree. For example child nodes of the different parent nodes. Shared bandwidth limit should constrain a total bandwidth that can be used by those nodes, similar to the bandwidth limit of the parent node restricting a total bandwidth available to all child nodes.
- For example, Node20 and Node23 can be configured to have a shared bandwidth limit and then the total amount of bandwidth available for these nodes should not exceed a bandwidth available to the node's parents Node10 and Node11, respectively and sum of the bandwidth used by Node20 and Node23 should not exceed shared bandwidth limit. A node cannot have assigned both regular bandwidth limit and shared bandwidth limit.

A.1.4 Scheduling Rules

To guarantee correct scheduling, the following rules must apply:

- Any tree node can be selected for scheduling traversal or scheduled only if it has non-zero bandwidth share, and it has not reached its maximum bandwidth limit. If one of this conditions does not hold, a node is considered to be blocked, and cannot be scheduled.



- Leaf node can be scheduled only if an associated set of transmit queues has work available for transmission. Leaf node is a tree node and bandwidth limit and bandwidth share constrains can be applied to it as to any other tree node as previously described.
- Parent node can be scheduled only if it has at least one child node that can be scheduled. Parent node is a tree node and subject to the bandwidth limit and bandwidth shared constrains described in the first bullet.

This set of rules allows to guarantee that a node is added to the scheduling traversal, only if a path exists from this node to non-blocked leaf node consisting of non-blocking nodes among the node descendants. This prevents false scheduling and makes a depth of the scheduling tree traversal equal to the depth of the tree.

A.1.5 Arbitration Schemes

The X710-TM4 supports several arbitration schemes that can be used to select a node among siblings during scheduling tree traversal. Each set of sibling nodes has an arbitration scheme assigned to it. Siblings are participating in arbitration only if they are non-blocked.

Each node might have two sets of credits associated with bandwidth allocation credits and bandwidth limit credits.

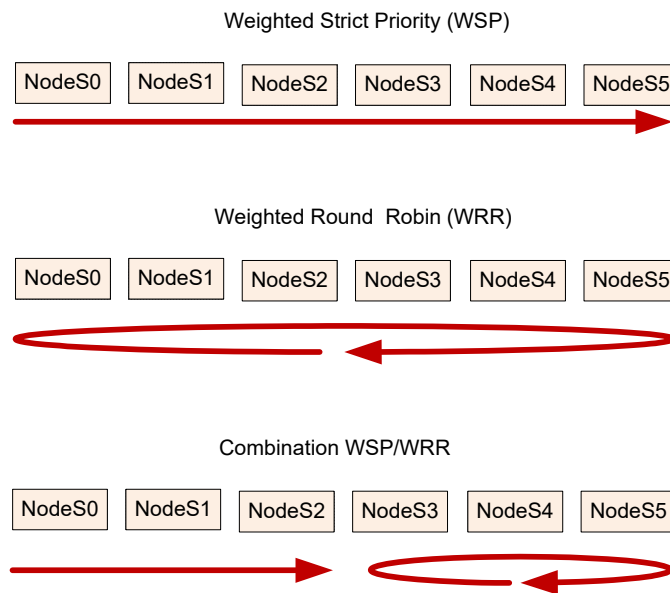


Figure A-3. Arbitration Options

- Weighted Round Robin (WRR). This arbitration scheme equally prioritizes nodes and gives every sibling node a chance to be selected in cyclic round robin fashion as long as it has credits and not blocked.



- Arbiter iterates sibling nodes in cyclic round robin fashion. Every scheduling cycle arbiter moves to the next sibling node, even if credits of the current node are not completely used. This allows even distribution of scheduling among sibling nodes as long as they have remaining scheduling credits. If the current node is blocked, then the arbiter moves to the next sibling node. If the arbiter used on all node credits, it replenishes node credits immediately, but does not use them until all non-blocked nodes had their credits used and replenished.
- Weighted Strict Priority (WSP). This arbitration scheme attempts to schedule nodes based on their priority as long as the node stays within its bandwidth allocation with respect to other sibling nodes and within its bandwidth limit.
- Sibling nodes are ordered accordingly to their priority. The arbiter always starts iteration from the first sibling node. If current node is blocked, the arbiter moves to the next sibling node. The arbiter effectively keeps scheduling a node with the highest priority as long as this node has credits and not blocked.
- Combination of Weighted Strict Priority and Weighted Round Robin. This arbitration scheme combines WRR and WSP schemes previously described. Sibling nodes should be organized accordingly to the arbitration scheme: WSP nodes first, followed by WRR nodes. Each node can be scheduled using one arbitration scheme only. WSP nodes are ordered based on their priority. Each scheduling cycle arbiter should attempt to schedule a node from WSP nodes using WSP arbitration scheme described above. If all WSP nodes are blocked, arbiter should attempt to select a node from WRR nodes. For WSP nodes arbiter should always start arbitration from the first node, and for WRR nodes, it should always start arbitration from the last scheduled node.

A.1.6 Shared Bandwidth Limit

The bandwidth limit attribute described in [Appendix A.1.3](#) can be configured for individual switching component, VSI or TC/UP. Some usage models imply instantiation of the multiple tree nodes for the same system resource, such as VEB or VSI. In order for the system resource be bandwidth limited, the transmit scheduler must support the shared bandwidth limit to multiple tree nodes. See [Appendix A.1.7.1](#) for an example of such configuration.

Not all scheduler configurations require use of the shared bandwidth limit. See [Appendix A.1.7.2](#) for an example of the bandwidth distribution scheme that does not use the shared bandwidth limit.

Instantiation of the shared bandwidth limiter involves instantiation of the dedicated rate limiter for tree node representing the same system resource. If all tree nodes use their dedicated credits, then no shared credits are accumulated. If some of the tree nodes do not use their credits (such as due to lack of work available), shared bandwidth limit credits are accumulated and other tree nodes representing the same system resource can take advantage of available shared credits. This scheme allows starvation free implementation of the shared bandwidth limits.

For example, if software instantiated shared rate limiter of 8 Gb/s and associated this rate limiter with VSI having 4 TCs configured, then each of VSI tree nodes can be configured with 2 Gb/s dedicated bandwidth limits. If all TCs of the VSI transmitting 2 Gb/s worth of data, then no shared credits are accumulated and VSI is limited to all its TCs to 2.5 Gb/s. If one of the TCs currently does not have data available for transmission, the other 3 TCs can take advantage of the available 2 Gb/s bandwidth and increase its bandwidth limit to 2.6 Gb/s.

Instantiation of the shared bandwidth limit and corresponding dedicated bandwidth limits is the responsibility of firmware and transparent to software. Depending on the configuration, firmware might translate a software request to instantiate a bandwidth limit to the system resource to instantiation of several dedicated bandwidth limits and one shared.



Dedicated bandwidth limits must be configured in such way that their total should add up to the shared bandwidth limit. Effectively shared bandwidth limit credits are distributed among dedicated bandwidth limits. Firmware can choose to distribute shared bandwidth limit equally or relatively to the bandwidth allocation.

The X710-TM4 implementation of the shared bandwidth limits allows each tree node use its dedicated credits even after periods of inactivity. Shared bandwidth limit has a control (max amount of accumulated credits) that allows to control burst generated by tree nodes associated with the same shared bandwidth limit.

A.1 The X710-TM4 Tree-based Scheduler

The X710-TM4 employs a centralized scheduling approach. All scheduling decisions are made by the main scheduler and followed by arbiters throughout the transmit path. The X710-TM4 scheduler schedules work for both stateless and stateful transmit queues.

The X710-TM4 stateless and stateful transmit descriptor queues are organized in queue sets; stateful and stateless queue sets, respectively. For the The X710-TM4 scheduler, each scheduling cycle selects one of queue sets. The index of a selected queue set is passed to the transmit processing logic. Selection of the transmit queue from the queue set is outside of the scope of scheduler definition and should be done by transmit processing logic.

Each leaf node in the scheduler tree is identified with a pair of queue sets. Selection of the leaf node is done using scheduling tree traversal, as described in [Appendix A.1.2](#). Once leaf node is selected, the scheduler selects one of two queue sets to be served at the current scheduling cycle. Arbitration between queue sets is round robin, one scheduling request in a time. If queue set does not have work available, a second queue set is chosen instead. One of the queue sets must have work available to comply with scheduling rules described in [Appendix A.1.4](#).

With each selected queue set, the scheduler specifies a quanta meaning the number of bytes that can be transmitted per scheduling cycle. Quanta can span multiple transmit queues and multiple descriptors. It is up to the transmit processing logic to decide how many descriptors and queues to use. Queues assigned to the same queue set have the same priority and arbitrated round robin by processing logic. The X710-TM4 supports one quanta that can be configured to values in range from 4 KB to 64 KB. Software should be advised to keep the quanta value low in order to keep burstiness of traffic generated by the X710-TM4 low. This is particularly important for the bandwidth limited traffic. Once queue set has been selected, a quanta worth of traffic for this queue set would be generated at wire-speed regardless of the queue set or scheduling path bandwidth limit. An increase of quanta is only recommended to resolve potential performance deficiencies cause by a small quanta value.

The X710-TM4 hardware does not associate the physical identity with a scheduling tree node. Thus each node can be configured to represent any physical resource.

To allow proper support of PFC, each node carries a bitmap of TCs that can be currently scheduled through this node. For example, there is at least one leaf node among node descendants that belongs to that TC and has transmit work available and not blocked. The X710-TM4 scheduler uses a bitmask register allowing to mask off TCs that are blocked by PFC and prevent them from scheduling.

Note: Though the scheduler allows top level of arbitration to be UP, PFC it is still performed on the TC level and therefore UPs mapped to the same TC are flow controlled together.

Queue set can be scheduled only if one of the transmit queues belonging to the set has work available. For example, descriptors are posted by software and ready for transmission.



A.1.7 Basic Scheduling Tree Configuration Options

The X710-TM4 scheduler is based on the abstract scheduling tree concept described in the [Appendix A.1.1](#). Structure and configuration of the scheduling tree depends on the system configuration and expected bandwidth distribution model. It is configured using scheduler configuration tables described in the [Appendix A.2](#). Those tables define a tree topology, bandwidth allocation, arbitration scheme and bandwidth limit for each of the tree nodes.

Abstract scheduling tree structure, described in the [Appendix A.1.1](#), allows abstract representation of various complex chip configuration. Each tree node can be associated with a physical resource, such as S-comp, VEB/VEPA, VSI, TC and UP etc. Tree structure is very convenient to represent hierarchical relationship and dependency of various chip resources and allows scheduling process to naturally follow the chip resource hierarchy. Use of abstract tree structure, when tree nodes can represent different chip resources, allows great flexibility, and ability to support wide variety of system and chip configurations without creating dedicated solution for each one of them.

The X710-TM4 supports a variety of the scheduling tree configurations. Two main configurations are enabled by the general purpose programming interface (those are primarily discussed here). Alternative configurations can be enabled using a privileged programming interface.

Primary, the two scheduler configurations modes are: ETS-based and VNet-based configurations. Configuration option selection is done on a per physical port base. It is programmed by firmware using TBD register and kept in NVRAM.

A.1.7.1 ETS-based Scheduler Configuration

[Figure A-4](#) shows a logical diagram depicting a concept of the ETS-based bandwidth distribution scheme.

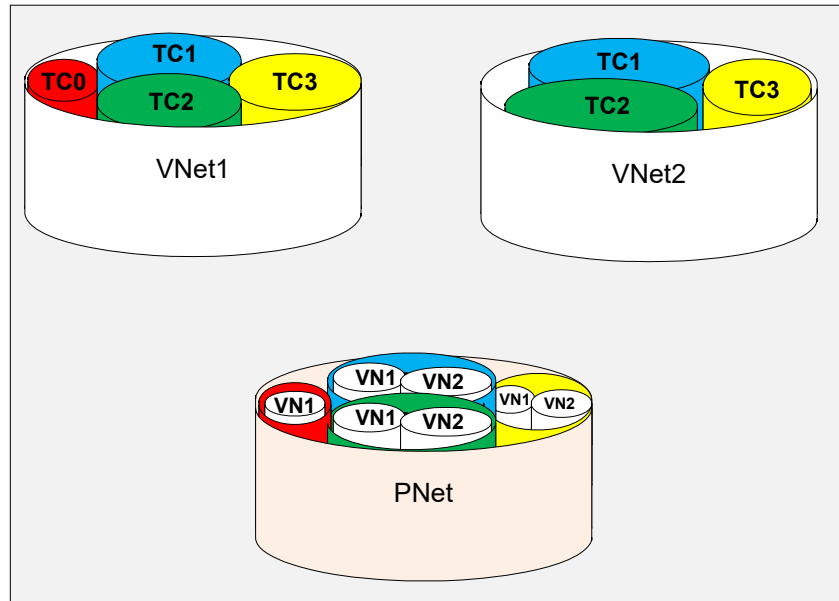


Figure A-4. ETS-based Bandwidth Distribution

Figure A-4 shows two virtual networks VNet1 and VNet2. Each virtual network carries traffic of different types by Traffic Class (TC). Both virtual networks are merging into a physical network. A physical network is configured accordingly to the ETS specification. Bandwidth of a physical network is divided between TCs. Each TC carries traffic belonging to their respective traffic types of both virtual networks. Bandwidth distribution between virtual networks within same TC is invisible outside of the device.

Figure A-4 shows a physical network concept of the bandwidth distribution in the ETS-based configuration. Bandwidth is first distributed between types of traffic (TCs) and then it is distributed between virtual networks within each TC. Therefore if one of the virtual networks does not have traffic of the certain type currently available, the remaining bandwidth would be consumed by the same traffic type from other virtual networks. For example, if VNet2.TC3 does not have any work available, the remaining TC3 bandwidth is consumed by VNet1.TC3, as long as it has enough work available and does not exceed any bandwidth limit.

This scheme allows bandwidth allocation per traffic type for each switching component and VSI and does not support a bandwidth allocation per virtual network.

This is a default bandwidth allocation scheme for the X710-TM4 scheduler.

Figure A-5 shows an example of the system configuration and bandwidth allocation using ETS-based scheduler configuration scheme.

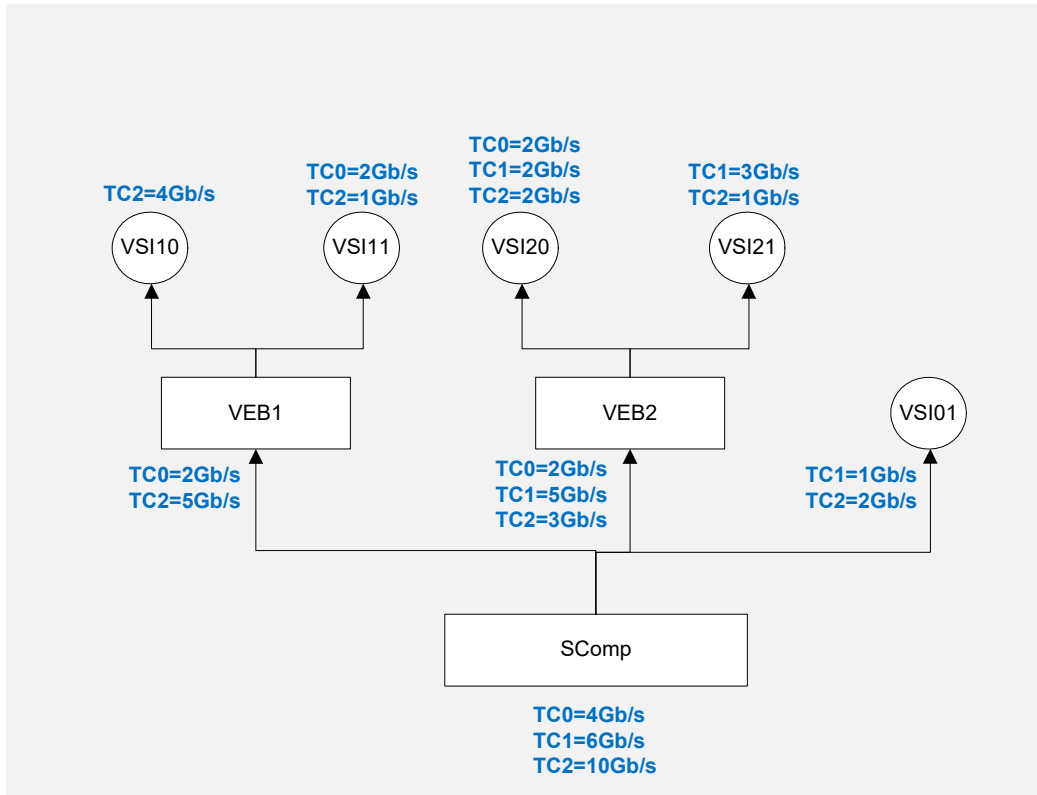


Figure A-5. Example of ETS-based System Configuration

Each one of the switching components (S-comp, VEB1, VEB2) and VSIs can be configured with ETS (bandwidth allocation per traffic type or TC) and bandwidth limit. Bandwidth allocation must be consistent, such as if a TC has certain bandwidth allocated on the egress S-comp port, the sum of the bandwidth allocated to the same TC on each level of tree hierarchy should be equal to the bandwidth allocated on S-comp egress port. For example, if S-comp.TC0=4 Gb/s, then $VEB1.TC0 + VEB2.TC0 + VSI01.TC0 = 4 \text{ Gb/s}$ and $VSI10.TC0 + VSI11.TC0 = VEB1.TC0$.

Figure A-6 shows an ETS-based scheduling tree configuration corresponding to the system configuration previously shown.

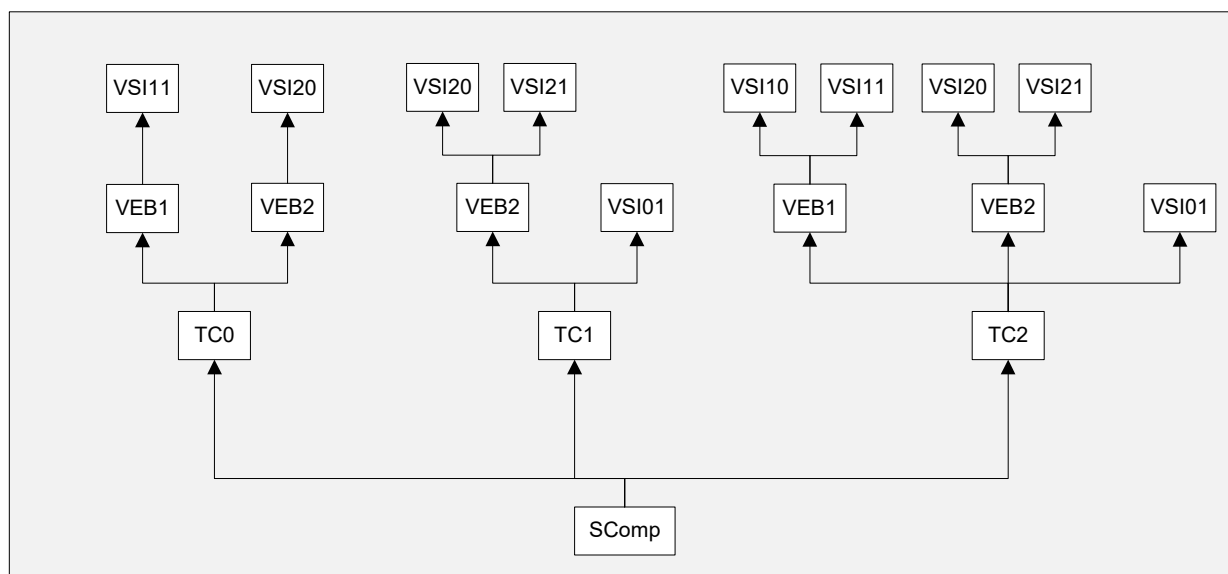


Figure A-6. ETS-based Scheduling Tree Configuration Example

Structure of this scheduling tree resembles topology of the system configuration shown on [Figure A-5](#). This tree has four tree levels (one not shown):

- TC level is the lowest tree level. It is configured with a physical port or S-comp egress port ETS bandwidth allocation (such as nodes TC0, TC1 and TC2 shown in [Figure A-5](#)).
- UP level (not shown in [Figure A-5](#)) is the next tree level. It also follows the S-comp egress port ETS configuration. If more than a single UP is associated with a TC, this level defines a bandwidth allocation among UPs associated with the same TC.
- Switching component level is a next scheduling tree level. This level consists of switching components (VEB/PA) and VSIs directly attached to S-comp. Each tree node corresponds to the TC/UP configured for the switching component/VSI. Single switching component and VSI might appear in the tree multiple times, once for each configured TC/UP. For example, nodes VEB1_TC0, VEB1_TC2, VEB2_TC0, VEB2_TC1, VEB2_TC2, VSI01_TC1 and VSI01_TC2 shown in [Figure A-5](#).
- VSI level is the last tree scheduling level. This level consists of the VSIs of switching components. Each tree node corresponds to the TC/UP configured for the VSI. A single VSI might appear in the tree multiple times, once for each configured TC/UP. For example, VSI10_TC2, VSI11_TC0, VSI11_TC2, VSI20_TC0, VSI20_TC1, VSI02_TC2, VSI21_TC1 and VSI21_TC2.

An internal scheduling tree configuration is not visible to software. Software sees a logical bandwidth configuration shown in [Figure A-5](#) and controls exposed to software relate to the switching elements and VSIs shown on [Figure A-5](#). Firmware creates and manages the internal scheduler configuration tree and translates software controls to the actual scheduler configuration. For example, when software provides an ETS configuration of VEB1, it lists TCs enabled for VEB1 (TC0 and TC2) and specifies an ETS bandwidth allocation per TC. Firmware allocates multiple scheduling nodes for VEB (VEB1_TC0 and VEB1_TC2) and configures those nodes with credits provided by software.

The scheduling tree can be extended with more scheduling levels. The maximum depth is the tree supported by hardware is seven.

Software can enable a bandwidth limit to any switching component and VSI. If a switching component or VSI has more than single TC enabled, it appears multiple times in the transmit scheduling tree. To configure a bandwidth limit to a switching component or VSI, firmware must use a shared bandwidth limits described in [Appendix A.1.6](#). Use of shared or basic bandwidth limits is hidden from software and a decision whether to use basic or shared bandwidth limit is done by firmware depending on the selected bandwidth allocation mode and system configuration.

A.1.7.2 VNet-based Scheduler Configuration

Figure A-7 shows a logical diagram depicting a concept of ETS-based bandwidth distribution scheme.

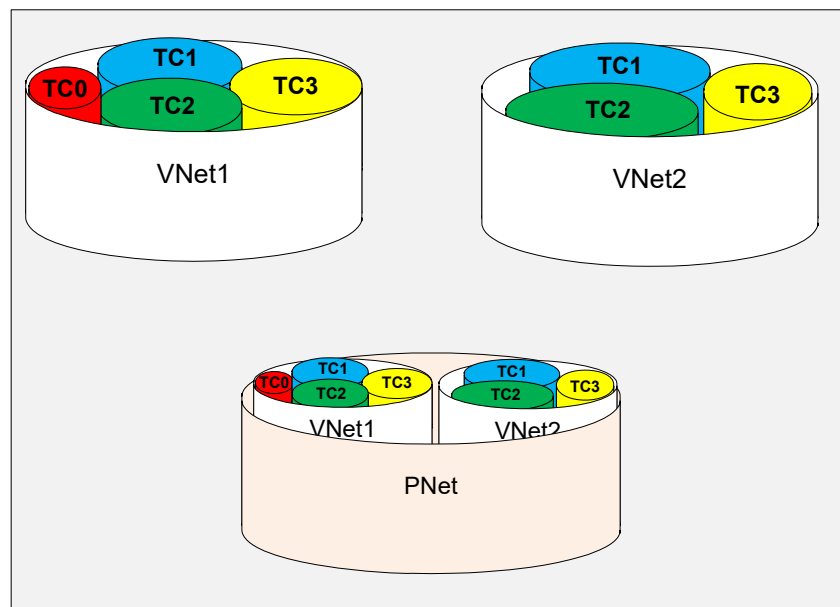


Figure A-7. VNet-based Bandwidth Distribution

Figure A-7 two virtual networks VNet1 and VNet2. Each virtual network carries traffic of different types by TC. Both virtual networks are merging into a physical network. A Physical network is configured to distribute bandwidth between virtual networks first and then distribute bandwidth within each virtual network between traffic types configured for that virtual network.

This bandwidth distribution scheme seems to be logical for the cloud-based environments with multiple tenants willing to share physical resources, providing isolation of bandwidth allocation throughout the fabric. Current standards do not allow taking advantage of this bandwidth distribution scheme.

This scheme allows a distribution bandwidth between virtual networks and provides global SLA guaranties as well as not limit those to the particular traffic type, as ETS-based scheme described in [Appendix A.1.7.1](#). Within each virtual network, bandwidth can be then be distributed between types of traffic. For example, if VNet2.TC3 does not have any work available, the remaining bandwidth is consumed by other traffic types configured for VNet2 (VNet2.TC1 and VNet2.TC2) as long as they have enough work available and do not exceed any bandwidth limit, rather than having VNet1.TC3 consuming it like in ETS-based scheme.

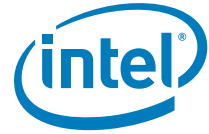


Figure A-8 shows an example of the system configuration and bandwidth allocation using VNet-based scheduler configuration scheme.

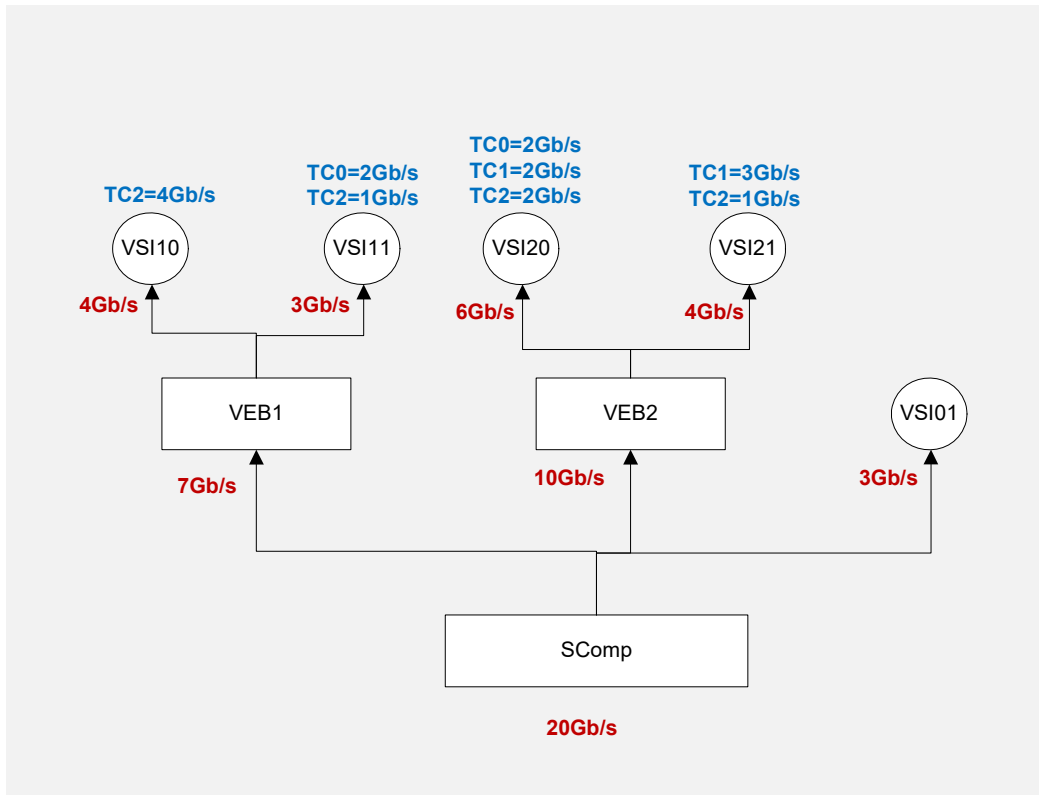


Figure A-8. Example of VNet-based System Configuration

Each of the switching components and VSIs has an allocated bandwidth and might have a bandwidth limit associated with it. VSIs are also configured with ETS or per traffic type bandwidth allocation. Bandwidth allocation must be consistent. The amount of bandwidth allocated to the switching component should be equal to the sum of bandwidth allocated to all VSIs and the amount of bandwidth allocated for a VSI must be equal to the sum of bandwidth allocated to all TCs. For example, bandwidth allocated to VSI10 (4 Gb/s) and VSI11 (3 Gb/s) should be equal to the bandwidth allocated to VEB1 (7 Gb/s).

Figure A-9 shows an ETS-based scheduling tree configuration corresponding to the system configuration previously shown.

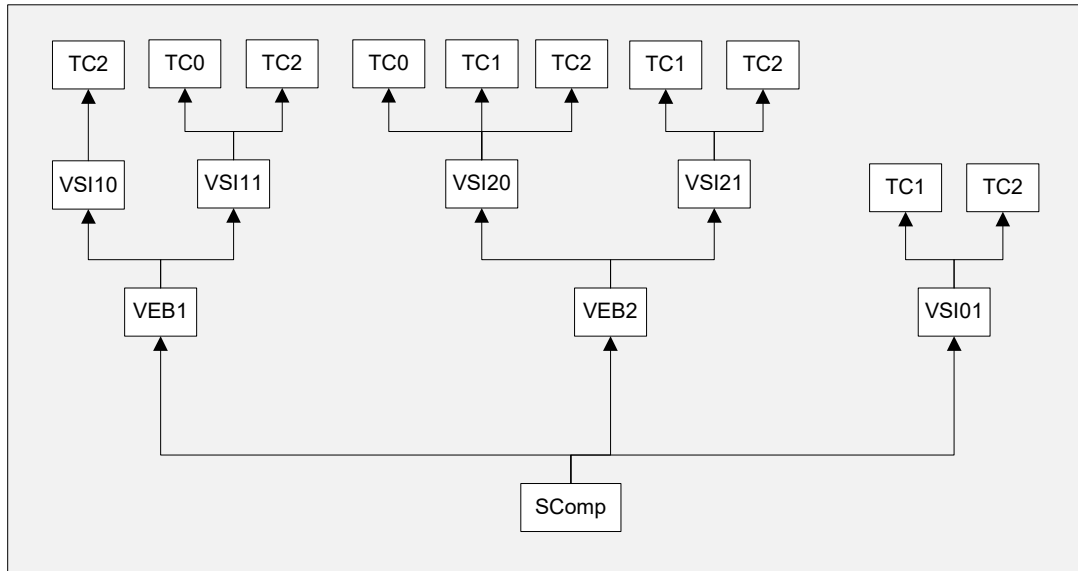


Figure A-9. VNet-based Scheduling Tree Configuration Example

The structure of this tree exactly follows a topology of internal switching fabrics shown in [Figure A-8](#). This tree has four levels (one not shown).

- Switching components and VSI tree level is the lowest tree level, programmed with bandwidth allocation for switching components and VSIs directly attached to S-comp (such as nodes VEB1, VEB2 and VSI01). Each VSI and VEB has a single node associated with.
- VSI tree level is the next tree scheduling level, programmed with bandwidth allocation for VSIs attached to the switching components (such as nodes VSI10, VSI11, VSI20, and VSI21). Each VSI has a single tree node associated with it.
- TC tree level is a first tree level that represents VSI ETS. This tree level exists for each VSI that has ETS enabled. This level is configured to distribute bandwidth between TCs of the same VSI.
- UP tree level is the second ETS level. It is not shown in the previous diagram. It is used to configure bandwidth allocation between UPs belonging to the same VSI TC.

Similar to the ETS-based configuration, software is not aware of an internal scheduler configuration tree structure. Software is using logical representation, shown on [Figure A-8](#) to program scheduler configuration tables. Firmware translates software AdminQ commands and performs the actual configuration of the scheduler configuration tables.

If software requests to instantiate a bandwidth limit to the switching component or VSI, firmware should use a basic bandwidth limits, since each switching component and VSI is represented by single tree node.



A.1.8 Alternative Configurations

This section gives examples of several alternative configurations that are slight modification of the default configuration described in [Appendix A.1.8](#).

A.1.8.1 VEB Directly Attached to a Physical Port

One of the possible device configurations, is omitting the S-comp switching component and connecting a single VEB or VEPA switching component directly to the physical port.

This configuration simply eliminates one of the tree scheduling levels. [Figure A-10](#) and [Figure A-11](#) show the possible tree configurations in ETS-based and VNet-based schemes.

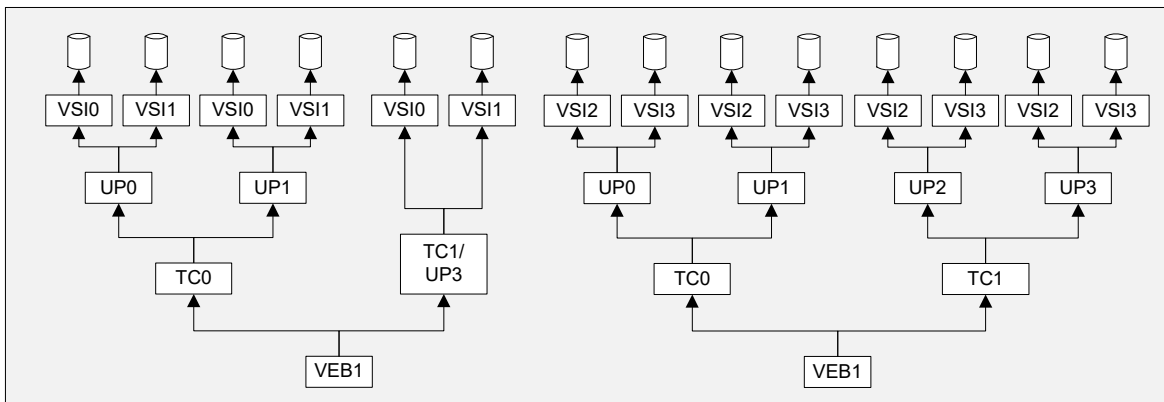


Figure A-10. VEB Direct Attached Scheduler Tree Configuration (ETS-based)

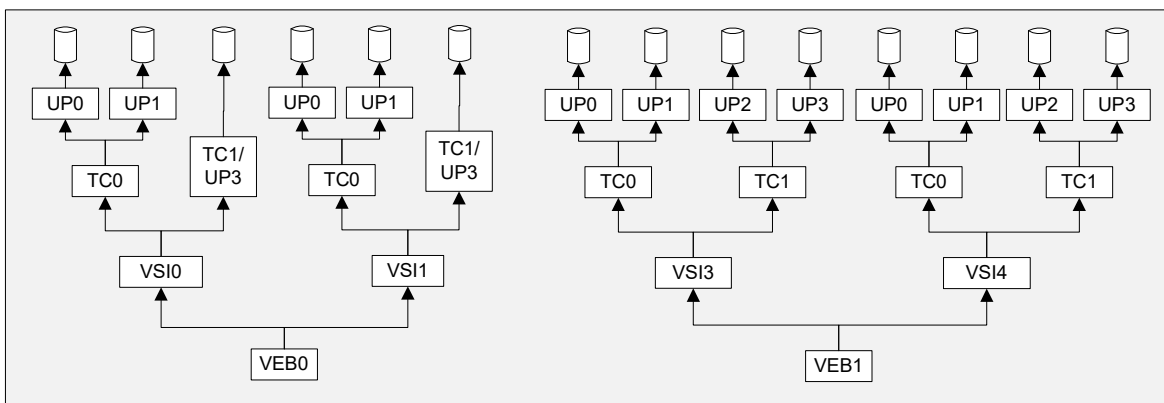


Figure A-11. VEB Direct Attach Scheduler Tree Configuration (VNet-based)

Figure A-10 and Figure A-11 show an example of the direct VEB attached configuration for two physical ports. Each port has one VEB attached to it, VEB0 and VEB1, respectively. Each VEB is configured with two VSIs. All VSIs have ETS enabled. Some of the TC nodes are collapsed since they have only one UP assigned to them (such as TC1 on the left side subtree). Figure A-10 shows an ETS-based tree configuration and Figure A-11 shows a VNet-based tree configuration. In both configurations, a VEB node takes a place of the root on the tree. The rest of the nodes are distributed based on the respective bandwidth distribution scheme.

A.1.8.2 Tree Level Extension to Reduce Number of Siblings

Scheduling performance depends on multiple factors, such as depth of the scheduling tree, number of sibling nodes and number of active bandwidth limiters.

The scheduling tree might be configured with a large number of siblings on different tree levels. Here a number of siblings estimate per tree level:

- S-comp tree level. Four siblings, as a number of S-comps or physical ports.
- VEB/VEPA/S-channel tree level. Eight VEB/VEPA nodes plus the maximum number of VSIs supported per physical port or since VEB/VEPA intermediate switches consume at least one VSI, the number is the maximum number of VSIs.
- VSI tree level. Maximum number of VSIs.
- TC and UP levels. Eight siblings per VSI (up to eight leaf nodes on two levels together).

Two of five tree levels have a potential to have a large number of sibling nodes. The scheduler performance depends on the depth of the tree and number of siblings. Firmware might be required to trade-off a depth of the tree and reduction in the number of siblings nodes to reduce an impact of the tree structure on the scheduler performance. This can be done by introducing an intermediate level of dummy tree nodes that do not carry any functionality beyond splitting nodes with large numbers of children to the two level subtrees.

Figure A-12 shows an example of such a tree modification.

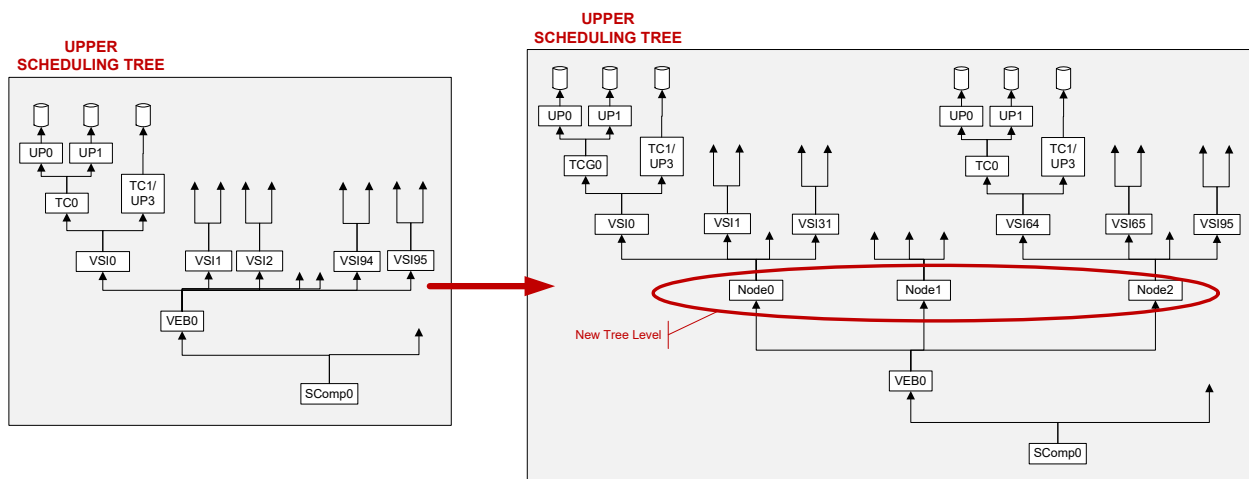


Figure A-12. Sibling Nodes Reduction Example



Figure A-12 shows an example of a tree with an excessive number of siblings on the VSI tree level (96 VSIs instantiated for the VEB0) on the left side and the new tree with reduced number of siblings on the right side. The new tree has one tree level added: Node0, Node1 and Node2. Those nodes are made children of the VEB0 node and each of the new nodes carry 32 children (VSI nodes). Bandwidth allocation for those nodes should be based on the bandwidth allocation for the child VSI nodes. New nodes should not have bandwidth limits assigned to them and the arbitration scheme used on this tree level should be WRR.

The new tree structure might cause some deviations to the relative bandwidth allocation among VSIs. For example, assuming that all 96 VSIs on the left side tree has identical bandwidth allocation, then if the only VSIs that have work available are VSI0, VSI32 and VSI64-VSI71 (a total of 10 VSIs).

- Left side tree configuration. Each VSI gets equal bandwidth allocation of 1/10 of bandwidth available for VEB0.
- However in the right hand side tree configuration, bandwidth distribution would be different. Since all VSIs are configured in the original tree with equal bandwidth allocation, in the new tree, Node0, Node1 and Node2 have even allocation as well. Also, given the same work available assumption, VSI0 and VSI32 gets 1/3 of the VEB bandwidth and VSI64-VSI71 gets 1/24 of VEB0 bandwidth allocation each.

In spite of deviation in relative bandwidth distribution shown in the previous example, it is not clear whether in steady state this deviation would be important enough to prevent firmware from re-balancing the scheduling tree to improve overall scheduler performance.

A.1.8.3 Multiple Queue Sets Directly Attached to a Physical Port

This section describes a non-standard device configuration that might be important for future customers. In this configuration, the X710-TM4 does not have any internal switching components instantiated. Instead, software needs to allocate a large number of stateless queues, assign those queues to the different physical ports and specify relative bandwidth allocation as well as bandwidth limits to those queues.

Figure A-13 shows a possible scheduling tree configuration:

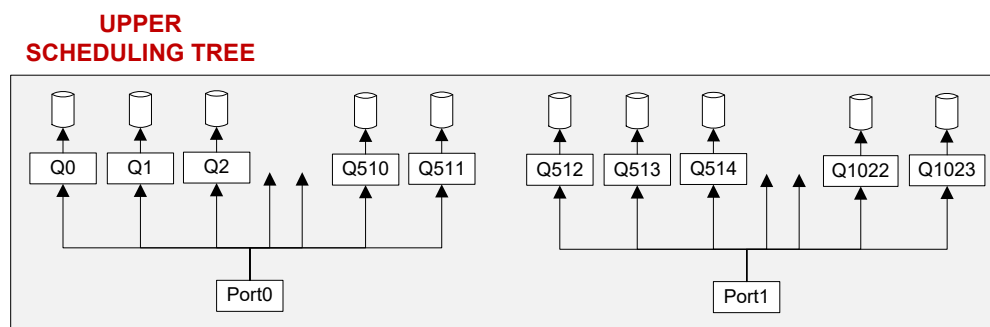


Figure A-13. Multi-queue Flat Tree Configuration

The X710-TM4 scheduler can support upto 1024 independently schedulable queues. Each queue is associated with a dedicated queue set. The scheduler tree structure is flat (two levels). The first level is a port arbitration level and second level is arbitration between queues belonging to the same physical port. [Figure A-13](#) shows an example of the dual physical port configuration with 512 queues assigned to each port. In this configuration the queue tree level has a large number of siblings. To reduce the number of siblings per tree level, the technique described in the [Appendix A.1.8.2](#) can be used.

[Figure A-14](#) shows a tree with reduced number of siblings per tree level.

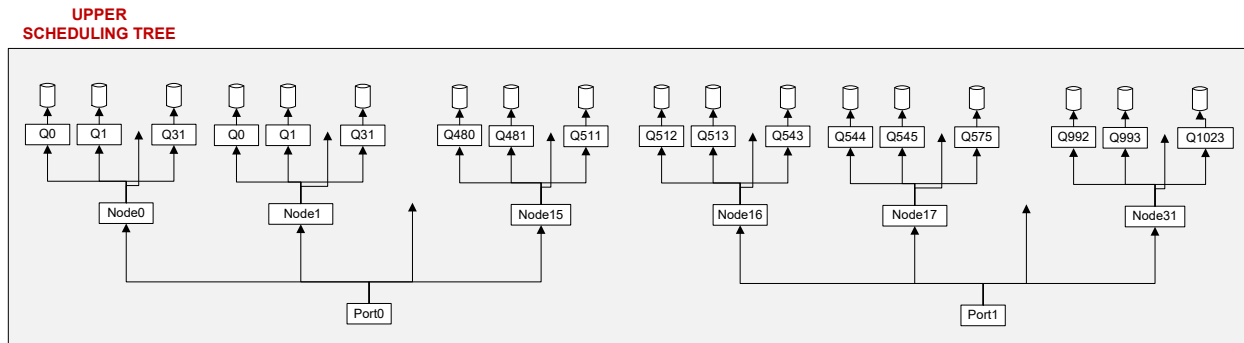


Figure A-14. Siblings Reduction

[Figure A-14](#) shows an alternative scheduling tree configuration, which has one more tree level added to reduce the number of sibling nodes. This new tree level consists of 32 nodes: Node0-Node31, respectively. Each node has 32 children queue nodes, which all together adds up to total of 1024 queues. All new nodes are configured based on the bandwidth distribution assigned to their child queue nodes. Reducing the number of siblings allows better control over the worst case scheduler performance and trades it for the additional tree level as well as possible deviations in bandwidth distribution as described in [Appendix A.1.8.2](#).

A.2 Scheduler Configuration

This section describes how to configure scheduling configuration tables and guides firmware through implementing AdminQ commands defined in [Section 7.8.4](#).

A.1.9 Overview of Scheduler Configuration Tables

Scheduler is configured using a set of scheduler configuration tables. Scheduler configuration tables are shared by all PCI functions and managed by firmware. This section provides a high level description of each table that needs to be configured. Scheduler table configuration is performed by firmware and software can use AdminQ commands described in [Section 7.8.4](#) to alter default scheduler table configurations.

Each node in the scheduling tree has an entry in each one of the tables described in the text that follows and each node has the same index for addressing all the scheduler data structures.



A.1.9.1 Node Table

Node table is a main scheduler configuration table. This table is used to define a scheduling tree structure. Each entry corresponds to the individual node in the scheduling tree. There is technically no root node in the tree; however, there is a set of nodes that define the base of each tree. These base nodes need to be consecutively located as the first nodes in the table (node 0 through node N).

A parent node holds an index of the first child node and last child node. All children nodes of the same parent (sibling) should be organized by up to eight nodes (aligned to the node index). Each node octet is located continuously in memory. Node octets of the siblings are chained in a linked list using a node octet table.

The node table is organized in groups of eight entries (0-7, 8-15, etc.) to optimize the sibling arbitration cycles. There can be more than eight siblings that would require multiple reads of linked nodes octets. For example, 12 siblings require two reads from the nodes table and one read from the node octet table. 19 siblings require three reads from the nodes table and two reads from the nodes octet table, etc.

In addition to the tree configuration, each node table entry carries a work available and status information required for the scheduling, such as per TC work available information, which is based on children status information for parent nodes or on availability of work in queue sets in case of leaf node. An indication whether or not a node can be scheduled based on bandwidth limit and bandwidth allocation credits is available. A node table entry also used to keep bandwidth allocation credits for both relative and best effort bandwidth allocation schemes.

The number of entries in the node table is calculated based on the maximum number of leaf nodes supported. For the 1024 leaf nodes, assuming the default scheduling tree configuration nodes table should carry at most:

- 1024 nodes for UPs or TCs with single UP
- 512 nodes for TCs that have more than a single UP associated with them
- 384 nodes for VSIs
- 16 nodes for VEB/VEPA switching components
- 4 nodes for S-comps
- Total of $1024+512+384+16+4 = 1940$ tree nodes

Note: The maximum number of nodes in a tree configuration, assuming collapsing of nodes with single child is limited by the binary tree. The number of nodes in the binary tree that has 1024 leaf nodes is: $1024+512+256+128+64+32+16+8+4+2+1 = 2047$, therefore 2048 nodes should be sufficient to implement any tree with 1024 leaf nodes.

A.1.9.2 Branch Table

This table has same number of entries as a node table. Each entry in this table carries a list of nodes that are leading from the root node of the tree to the node. This table is a service table, and it allows traversal of the scheduling tree in the reverse direction (from child node to parent node). Keeping branch nodes in a separate table, rather than using a back-pointer from the child to parent allows pipelined backward tree traversal.

Branch table entry must be allocated for every tree node.



A.1.9.3 Bandwidth Limit Table

A bandwidth limit table carries a bandwidth limit configuration for each node. If bandwidth limit is enabled for the node, the corresponding entry in bandwidth limit table should be initialized with the credit update and the maximum number of bandwidth limit credits that can be accumulated for the node. The maximum number of credits is specified a number of quanta worth of credits that can be accumulated. Credits are specified in units of 8 bytes. Once a bandwidth limit table entry is configured, corresponding entry in the node table should be set to indicate that bandwidth limit is enabled for the node.

If bandwidth limit is not enabled for the node, then a corresponding bandwidth limit table entry should be initialized with zeros.

A.1.9.4 Shared Bandwidth Limit Table

A shared bandwidth limit table carries accumulated shared bandwidth limit credits. The number of entries in this table is limited to 512. Each entry can be associated with multiple bandwidth limit table entries. If a bandwidth limit table entry is associated with the shared bandwidth limit, it carries an index of the corresponding entry in the shared bandwidth limit table.

A shared bandwidth limit table carries a number of accumulated credits and the maximum number of credits that can be accumulated by that shared bandwidth limit.

A.1.9.5 Best Effort Table

This table is used to allow best effort scheduling for the siblings nodes. A nodes table entry of the parent node carries an indication of the bandwidth allocation scheme used for the children nodes. Two options are available: relative bandwidth allocation and best effort or zero-bandwidth guarantee allocation.

- In relative bandwidth allocation, each node gets a non-zero bandwidth allocated to it and the remaining bandwidth is shared across siblings with work available proportionally to the original bandwidth allocation.
- In a best-effort or zero-bandwidth allocation scheme, a node might have a zero bandwidth allocated to it. Those nodes won't be able to use any bandwidth until all nodes used their non-zero bandwidth allocation and then remaining bandwidth is equally shared among all nodes.

If relative bandwidth allocation is configured for the sibling nodes, then entry in the zero-bandwidth guarantee table corresponding to the parent node should be zeroed.

If best-effort bandwidth allocation is configured for the sibling nodes, then parent node should be configured with the total number of bandwidth allocation credits available for all children nodes.

A.1.9.6 Ready List Mapping Table

This table specifies mapping of each queue set to the corresponding leaf node index. This mapping is required to allow an unbalanced scheduling tree configuration and satisfy the requirement of continuous location of the sibling node octets in the scheduling tree. In addition to the index of the corresponding leaf node in the tree, this mapping table carries a TC of that queue set, and identification of the PCI function that owns the queue set.



A ready list mapping table also carries an index of the branch table entry to be used to traverse predecessor nodes of the leaf node associated with the ready list mapping table entry. Firmware can use this field to set an index of the first child entry in the branch table to all siblings to reduce the amount of branch table updates required for tree restructure.

Note: The X710-TM4 restricts each queue set be owned by one PCI function and carry traffic of a single TC.

Note: A ready list mapping table is the only table that is not ordered by the node indexes in the tree.

A.1.9.7 Nodes Octet Linked List Table

To simplify management of the nodes table, all nodes table entries are allocated in 1/8s. All nodes located in the same nodes octet are continuously located in the memory.

If parent node has multiple child nodes, these nodes should be located continuously within an allocated nodes octet. If the number of child nodes exceed a space available in one nodes octet, multiple nodes octets can be allocated. First nodes octet is continuously occupied by siblings starting from the first child node. First child node is recommended to be a first node in nodes octet, but not required. All subsequent siblings must be continuously located in linked nodes octets.

Child nodes belonging to different parent node are allowed to share nodes octet, but they must remain continuous with respect to other child nodes belonging to the same parent node.

A nodes octet table has 256 entries of 8 bits each. Each entry is an index of the next nodes octet in the chain. A nodes octet table entry is only valid if a corresponding nodes octet is included into the nodes octet chain or linked list and is not a last nodes octet in the list. A nodes octet index is calculated by node index >> 3. Hardware uses a first and a last node index to identify a first nodes octet and a last nodes octet in the linked list.

A.1.10 Scheduler Configuration Restrictions

This section describes set of restrictions on configuration of the X710-TM4 scheduler tables.

- Node indexing in scheduler configuration tables
- Majority of the configuration tables keep the same node index to allow easy access to information related to the same tree node
- Location of the base tree level nodes.
- Since tree root node does not carry any practical functionality, the zero level nodes must be continuously located in the beginning of the scheduler configuration tables and occupy entries 0-(N-1), where N is a number of the nodes on the base tree level. If tree root node is used with a bandwidth limit entire tree, this node still must be located at the beginning of the scheduler configuration table at entry 0.
- Special meaning for the Node0. Node0 cannot be a leaf node and cannot be a valid child node. Most of the time Node0 would be occupied by the node corresponding to the Port0. Therefore an index 0 can be used as an invalid node Index when referring to the Leaf or child nodes.
- Location of sibling nodes
- Scheduler configuration tables (all except for branch and ready list mapping) are organized and managed in node octets. Siblings must be continuously located within a nodes octet. If not, all sibling nodes can fit in the same octet and multiple octets can be used. If siblings consume multiple



node octets, they must be continuously located within nodes octets and nodes octets chained using node octet table. First child node is not required to be a first node within nodes octet.

A.1.11 Scheduler Resources Reduction

There are several steps that allow reduction in device resources. In particular, memory used by the scheduler.

- Reducing the number of stateful queue sets.
- POR defines the number of queue set pairs to match the number of leaf nodes. The X710-TM4 supports asymmetric stateful and stateless resource allocation. For example, the number of VFs supporting iWARP is limited to 32, while the total number of VFs is 128. Given this asymmetry, users can change a scheme of the queue set resource allocation and reduce the number of queue sets available for the stateful queues to 256. This change should not require modification of the scheduler configuration structures, since it already uses cross reference to associated leaf nodes and queue sets.
- Reducing the number of bandwidth limited nodes.
- POR defines that each tree node might have a bandwidth limit enabled. This requires a bandwidth limit table supporting a maximum number of tree nodes. An alternative approach would be to limit the number of tree nodes that can be bandwidth limited to 256 and add a cross reference between a nodes table and bandwidth limit. This reduces the size of the bandwidth limit table and relaxes performance requirements of the bandwidth limit update logic.
- Reducing the number of best effort sibling sets.
- POR defines that each node might be a parent node with children nodes scheduled using a best-effort bandwidth allocation scheme. An alternative approach would be to reduce the number of such nodes to 256 and add a cross reference to the nodes table, referring to the corresponding entry in the best effort table. This reduces the size of the best effort table.
- Elimination of the branch table.
- A branch table allows fast retrieval of the tree branch nodes and it is used to speed up updates that are done independently from the scheduling flow. Such as work available, suspend, bandwidth limit, credits return, etc. Instead of using a branch table, a nodes table entry can carry a back reference to the parent node. This allows backward branch traversal, but serializes the update operation and potentially causes performance degradation for the scheduler.

A.1.12 Firmware Scheduler Interface

Firmware uses a register-based interface to access and program scheduler configuration tables. In normal operation mode those registers are accessible and controlled by firmware only. In debug mode all scheduler interface registers are mapped to the PCI address space and can be accessed by other CPUs or host software. No access synchronization mechanism is provided by hardware, therefore in debug mode it would be up to software and firmware to coordinate their access to the scheduler programming interface.

The scheduler provides two types of accesses to its configuration structures:

- Immediate Access. A basic operation allowing to read and write one entry of the scheduler configuration tables. Usually firmware performs one table access in a time using this interface.



- **Batched Access.** A more advanced operation mode, allowing firmware to post multiple commands and have them executed as a single atomic operation. The scope of commands that can be used for batch operation is more limited than immediate commands and primarily includes operations like copying an existing entry to a new location or update a field in the table entry.

Firmware is allowed concurrent use of both interfaces. The scheduler does not guarantee ordering between Batched and Immediate commands, and executes them in round-robin sequence along with other scheduling operations.

Table A-1 lists all scheduler interface commands supported via the immediate or batched interface. The majority of commands can be posted via either one of interface. Any invalid command leads to a critical error and suspends a respective interface. Detected errors are reported in the TSCDIFSTATUS register. The interface can be enabled back by setting a *ICMDCLRERR* or *BCMDCLRERR* bit in the TSCDIFCTRL register.

Table A-1 Scheduler Interface Commands

Command Name	Command Attributes (Fields in Registers)		Description
	IFICMDL IFBCMDL	IFICMDH IFBCMDH	
WriteField	TBLTYPE, TBLENTYIDX	FLDOFFS, FLDSIZE, VALUE	Change a value of the specified table entry. Firmware provides a table name, index of the entry in the table, field offset within the entry in bits, size of the field and value to be written to the field. This command can be used both as an immediate or batch command.
ReadEntry	TBLTYPE, TBLENTYIDX	TBLTYPE, TBLENTYIDX	Write to the specified table entry with the data from the TSCDIFDATA register. The scheduler interface provides only one data register shared for read and write operation as well as for immediate and batch commands. Firmware provides the name of the table and index of the entry in the table. This command is mostly used for immediate operations, unless firmware chooses to update multiple table entries with the same data.
ReadEntry	TBLTYPE, TBLENTYIDX	TBLTYPE, TBLENTYIDX	Read from the specified table entry to the TSCDIFDATA register. The scheduler interface provides only one data register TSCDIFDATA shared for read and write operation. The scheduler interface provides only one data register shared for read and write operation as well as for immediate and batch commands. Firmware provides the name of the table and an index of the field in the table. This command is usually used for immediate operation only.
CopyEntries	TBLTYPE, TBLENTYIDX	NUMENTS, ENTRYIDX	Copy specified number of entries from source table entry index in the table and to the target entry index in the table. The maximum number of entries that can be copied with single operation is eight. Copy operation should not cross node octets both on source and target. If firmware attempts to copy more than eight entries or copies entries crossing node octet boundary, it should use multiple Copy Entries commands to accomplish that. Note that copying multiple entries in one command is allowed only in a node table and bandwidth limit table. For all other table types, NUMENTS must be set to one. Source and destination are not allowed to overlap. Firmware provides a table name, indexes of source and target tables entries and number of entries to copy. This command is mostly used for batched operation, but can be used for immediate operation as well.



Command Name	Command Attributes (Fields in Registers)		Description
	IFICMDL IFBCMDL	IFICMDH IFBCMDH	
CopyField	TBLTYPE, TBLENTYIDX	FLDOFFS, FLDSZ, ENTRYIDX	Copy a specified field from the source table entry to the target table entry. Firmware provides a table name, indexes of the source and the target table entries, field offset within an entry in bits and size of the field. This command is mostly used for batched operation, but can be used for immediate operation as well.
CopyAndShift	TBLTYPE, TBLENTYIDX	ENTRYIDX	This command applies to branch table only. It allows copying a source entry to the target entry and shift content of the target entry by one field. For example, predecessor0 in source entry becomes predecessor1 in the target entry, etc. This operation is useful when creating a child branch table entry from the parent branch table entry. Firmware provides indexes of the source and target branch table entries. This command is mostly used for batched operation, but can be used for immediate operation as well.
Control	CTRLTYPE,TBLTY PE,TBLENTYIDX	ENTRYIDX	Controls batch interface. Currently the only supported control commands are a node BatchDone, Suspend and node Resume commands. The BatchDone command completes a batch and allows the scheduler to switch to perform other scheduling flows. This command is used for batched operation only. Node Suspend/Resume commands allows firmware to suspend/resume specified node or ready list. The transmit scheduler will be setting/clearing a Suspend bit in the nodes table or in ready list mapping table entry, depending on the table type specified in the request and will be updating a work available status of the branch nodes, respectively. If a suspended/resumed node is the only node with work available for the particular TC, parent node will have its work available status updated as well as change propagated to other predecessor nodes. Node Suspend and Resume commands can be used both as Batched or Immediate Commands,.

Submission of Immediate and Batched commands is performed using the TSCDIFICMDH/TSCDIFBCMDH and TSCDIFICMDL/TSCDIFBCMDL registers. Only a few commands are using the TSCDIFICMDH/TSCDIFBCMDH registers (WriteField, CopyField and CopyEntries). If a command does not use the TSCDIFICMDH/TSCDIFBCMDH register, it can be posted just by writing to the TSCDIFICMDL/TSCDIFBCMDL register. Otherwise, firmware writes to the TSCDIFICMDH/TSCDIFBCMDH register first and follows it with a write to the TSCDIFICMDL/TSCDIFBCMDL register. A write to a low register indicates to hardware that the command is ready and can be processed.

Both immediate and batched interface support a limited number of outstanding commands [Immediate (single command) batch interface up to 63 commands]. Firmware must use the TSCDIFSTATUS register to validate that the interface has a free space for the new command. Overflow of immediate or batched interfaces causes critical error that is reported in the TSCDIFSTATUS register and leads to suspension of the interface.

A.1.12.1 Immediate Command Interface

The Immediate command interface is provided by the TSCDIFICMD, TSCDIFDATA and TSCDIFSTATUS registers. This interface is mostly used for WriteEntry and ReadEntry commands.

Firmware flow usually includes:

- Read the TSCDIFSTATUS register to make sure that no Immediate command is pending.
- This operation can be skipped if previously posted immediate operation has completed.



- Write data to the TSCDIFDATA register (for WriteEntry operations). Data should be written to the lower words of the TSCDIFDATA register. Order of writing data is not important.
- Write a command to the TSCDIFICMDH/L registers. Only few commands are using the TSCDIFICMDH register, if posting different command, firmware can skip writing to the TSCDIFICMDH register and fill the TSCDIFICMDL register. Note that only the content of the TSCDIFICMDH register should be ignored.
- Read data from the TSCDIFDATA register for ReadEntry operations.

Though the immediate interface is primarily designed for ReadEntry and WriteEntry operations, it can be used by firmware to post other commands, except for Control commands.

A.1.12.2 Batched Command Interface

The batched command interface allows firmware to post multiple commands that need to be executed in atomic fashion. This interface allows firmware to perform a majority of scheduler configuration tables without suspending normal scheduler operation flow, except for execution of a series of batched commands, which consumes few scheduling cycles at most.

The batched command interface allows firmware to post up to 63 outstanding commands. Each sequence of commands posted to the batched interface should be completed with the BatchDone Control command posted to batched interface. This command indicates that the batch sequence completed and the scheduler can switch to perform other scheduling flows. Commands posted to the batched interface are not executed immediately, they are queued to the 63-deep batch FIFO. Firmware should ring DB by writing to the TSCDIFCTRL register to request execution of the posted sequence of batched commands.

Here is an example of posting the batched command sequence:

- Read the TSCDIFSTATUS register to make sure that the batched interface has enough space available.
- Write one or more commands to the TSCDIFICMDH/L registers. The TSCDIFICMDH register should be used for WriteField, CopyField and CopyEntry commands only.
- Complete the sequence by writing the BatchDone command to the TSCDIFICMDL register.
- Write to the TSCDIFCTRL register to ring DB.

Firmware is allowed to interleave commands posted via the immediate and batched interface. One usage model of that would be gathering information using ReadEntry immediate commands to build a sequence of Batched commands.

Firmware is allowed to post multiple sequences of commands to the batched interface, each terminated by the BatchDone command. Each time firmware posts the BatchDone command to the batched interface, it must ring DB. Hardware increments an internal counter with each DB ring and decrements with each processed BatchDone command. If the counter remains positive, hardware processes the next batch after it completes a round of performing other scheduling flows.

If the number of commands in sequence exceeds the size of the batch FIFO, firmware can still post that sequence. It needs to ring a DB after posting the first 63 commands, without posting BatchDone, keep posting more Batched commands monitoring the number of available Batched commands in the TSCDIFSTATUS register and after posting all commands concluded the sequence by posting BatchDone. In this case, the scheduler processes all commands in sequence atomically, but due to disparity in the scheduler and firmware performance characteristics, such an update can impact scheduler performance, since it won't be processing any other scheduling flows until it completes the batched sequence.



Firmware can use the batched command interface to post sequences of commands that not necessarily require atomicity.

A.1.13 Standard Scheduler Configuration using AdminQueue Commands

This section describes a logical updates of the scheduler configuration tables required to support standard AdmiQ commands defined in [Section 7.8.4](#).

A.1.13.1 Performance Requirements

Performance of the scheduler configuration update has two aspects:

- How quickly an individual update can be done or number of updates per second.
- How much a scheduler configuration update impacts scheduler performance or delays scheduling operations.

The first parameter of updates per second greatly depends on the firmware performance and the time it takes to complete the entire operation.

The second parameter depends on efficiency of the scheduler programming interface. Having the batched interface allowing firmware to build a sequence of operations that must be performed atomically with respect to other scheduling flows and have those batches executed by the scheduler in-between scheduling flows greatly reduces impact of the scheduling configuration updates on the overall scheduler performance as well as makes this impact independent of the firmware processing speed, as long as batched interface allows sufficiently long atomic sequence updates.

Based on the proposed scheduler update interface, and validated assumption that it supports sufficiently long atomic sequences, the impact of the scheduling configuration change for the standard scheduler configuration on the overall scheduler performance should be negligible.

Initial firmware evaluation based on the standard scheduler configuration changes and scheduler interface, leads to believe that firmware should be able to sustain a scheduler configuration update rate of 200 updates per second. If for some reason, the system requires an update rate beyond firmware capabilities, PF software issuing AdminQ commands are responsible to pace an update rate to match the rate supported by firmware.

A.1.13.2 Resource Allocation Strategy

This section focuses on configuration of the scheduler table depending on the selected scheduler configuration scheme (ETS-based or VNet-based). The majority of resource allocation recommendations are the same for both configuration schemes. When recommendations are different, it is explicitly indicated in the description.

The scheduler configuration tables allow dynamic and flexible resource allocation but they are not sized to provide the maximum amount of resources for each switching component. This precludes firmware from using a worst-case resource preallocation strategy. This together with restrictions on the scheduler tables configuration described in [Appendix A.1.10](#) requires firmware to adapt dynamic resource allocation scheme, described in this section, that might lead to potential subtree or entire tree reconfiguration to condense sparse allocated resources.

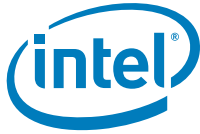


Most of the scheduler configuration tables are organized based on the tree node index, except for the ready list mapping table, which is organized by the queue set index and shared bandwidth limit table. For the allocation of the ready list mapping table see [Appendix A.1.13.3.1](#). For the allocation of the shared bandwidth limit table entry see [Appendix A.1.13.4.3](#).

In addition firmware might need to keep some extra tracking structures, such as a free list of node octets ordered by the number of free available entries in octet.

Following is a set of guidance that firmware should follow when allocating entries in scheduler configuration tables.

- Four lowest entries of the scheduling tree should be used for the switching components directly connected to the device's physical ports and should allow arbitration between ports. In a single-port configuration, only entry 0 of the node configuration table should be configured and other three entries should remain invalid. Firmware should preallocate the entire nodes octet (0).
- VNet-based Configuration. The X710-TM4 supports up to 16 VEB/VEPA switching components, with a maximum of 8 per physical port. Firmware should reserve 4 node octets, one per port for switching components or VSIs connected to S-channels. In an ETS-based configuration, VEB/PA nodes are distributed across multiple TCs and UPs and therefore might consume much more tree nodes.
- VNet-based Configuration. Due to a small number of switching components supported by the X710-TM4 (total of 20), firmware should not mix VSIs belonging to different switching components in the same nodes octet.
- VNet-based Configuration. Due to a relatively large number of TCs and UPs (the scheduler supports an average of two TCs/UPs per VSI), firmware won't be able to do such simplification for TC nodes and has to support TCs belonging to different VSIs sharing same nodes octet.
- ETS-based Configuration. TCs and UPs are the lowest tree level. The total number of combined TC/UP nodes is 8 per port. Firmware can pre-allocate those nodes (or even 16 nodes per port). POR enables single level ETS (TC level only). This reduces the number of nodes to be preallocated per physical port to 8.
- When allocating a new node, firmware should reuse nodes from already allocated octets belonging to the parent node, if any available. If not, a new nodes octet should be allocated.
- To reduce amount updates of a branch table, firmware should keep up to date only the branch table entries of the first sibling and configure both the ready list mapping table and rate limit table referring to the branch entry of the first sibling for all sibling nodes. This should reduce overall branch table update cost when adding a tree level or moving nodes within the scheduler configuration tables.
- When deallocating nodes, firmware should not deallocate a branch table entry for the first sibling of VSI or VEB nodes octet.
- Firmware should take advantage of extra nodes available (25% of available nodes). Note that the cleanup process can be postponed and performed in the background.
- Batched command interface, described in [Appendix A.1.12](#), should be used to allow atomic updates of the scheduler configuration tables without suspending scheduler operation.
- Though the batched interface allows atomic sequences longer than 63 commands, firmware should avoid using those due to hardware performance penalties. Overhead of use of long batched sequences should always be compared with overhead of suspending scheduler operation to perform scheduler configuration tables update.
- Guidances previously described do not necessarily apply to non-standard scheduler configuration supported by hardware. Hardware implementation should not assume that firmware always follows those guidances.



A.1.13.3 Queue Set Management

Firmware is responsible for the queue set allocation and management. This section describes queue set related firmware flows.

A.1.13.3.1 Queue Set Allocation

Firmware can either use a ready list mapping table or maintain its own private bitmap of allocated queue sets. Since a pair of queue sets is assigned to the same leaf node, both queue sets are allocated at the same time and it is sufficient to use a single bit or single entry in the ready list mapping table to represent both.

Though tree node configuration restricts allocation of the sibling nodes to consecutive entries in the node table, this restriction does not apply to the queue set allocation due to indirect referencing.

If firmware chooses to use the ready list mapping table, it should use a node Index 0 to indicate an invalid leaf node and use it as an indication of the vacant queue set. Once queue set is allocated, the leaf node index should be updated with the index of the tree leaf node.

When allocating a ready list mapping table entry, firmware should initialize all table entry fields.

A.1.13.3.2 Queue Set Identification

Queue set is identified by the queue set handle.

A queue set handle is provided upon completion of Create VSI, configuring VSI bandwidth allocation per traffic type, configuring VSI bandwidth limit per traffic type, or query VSI bandwidth configuration per traffic type. To retrieve a queue set handle, firmware needs to reach a leaf node referring to the ready list mapping table. In case of ETS-based configuration this can be done by walking tree nodes associated with VSI.

A queue set handle is 16 bits that carries the encrypted index of the queue set in the ready list mapping table.

A.1.13.3.3 Queue Set Deallocation

Queue set deallocation operation can be caused by an AdminQ command, directly or indirectly or by FLR/VFLR.

Usually queue set deallocation is a result of changing configuration of the corresponding VSI including disabling VSI.

Prior to initiating configuration changes leading to queue set deallocation, such as TC allocation to VSIs, or VSI deallocation, software must validate that all QPs associated with a queue set are destroyed and cannot lead to a work available update targeting this queue set. Firmware and hardware are not responsible to track resource allocation and trusts the PF software.

In case of VFLR or FLR, hardware schedules a special token to cleanup the pipeline. Once the hardware pipeline cleanup completes, firmware starts its own cleanup of the scheduling table entries owned by the VF/PF that were reset.

Hardware guarantees that no work is scheduled to a VF/PF under reset.

Deallocation of queue set is included into scheduler resources cleanup. Deallocated queue set is marked with leaf node of 0 and can be reused for other VF/PF.



A.1.13.3.4 Low Latency (LL) Queue Set

The VNet-based configuration scheme requires special handling of LL traffic. The scheduling tree is structured to distribute bandwidth between switching components and VSIs as well as performs TC arbitration only at the post VSI level. Such a structure might cause significant delays to the LL traffic, since it might take a round via entire tree before VSI is getting scheduled.

To overcome this configuration shortcoming and provide a priority scheduling for LL traffic within bandwidth allocated to the corresponding VSI, the X710-TM4 supports LL queue sets. Software can choose a particular queue set to be a LL queue set. This queue set should be associated with the LL (usually strict priority) TC.

Hardware uses a special priority arbitration scheme for such queue sets.

- With a LL queue set allocation, hardware requests the scheduler to preallocate bandwidth allocation credits using standard scheduling tree configuration.
- When work becomes available on the LL queue set and queue set has preallocated credits, its getting scheduled bypassing standard scheduling mechanism and uses preallocated credits.
- When the amount of preallocated credits falls below a configurable threshold, hardware requests allocation of additional credits.
- Allocation of credits is done using normal scheduling flow.
- If the LL queue set runs out of preallocated credits, then it is not scheduled using the priority arbitration scheme. This forces the LL queue set stay within configured bandwidth allocation limits.

Firmware when allocating a LL queue set, should set a *LL* bit in the ready list mapping table and post the command to the scheduler requesting to preallocate credits for that queue set.

A.1.13.4 Resource Tracking

Software uses VSI SEID and switching component SEID to refer to VSIs and switching components in AdminQ commands. The respective switching configuration table entries carry indexes of the corresponding scheduler tree nodes. Firmware is responsible to keep track and locate tree nodes associated with UPs and TCS and in case of ETS configuration per-TC VSI and switching component nodes.

The following sections discuss resource tracking in each configuration.

A.1.13.4.1 ETS-based Configuration

- UP and TC nodes.
- Firmware maintains a lookup table of 16 entries per physical port. Entries 0-7 allocated for TCs (TC0-TC7 respectively, entries 8-15 allocated for UPs (UP0-UP7), respectively. Each entry in the table carries an tree node Index corresponding to the TC/UP.

Note: Since POR has been reduced to enable single level ETS, firmware is required to track only 8 nodes per port or statically allocate those.

- VEB and VSI nodes.



- Firmware maintains a lookup table (VSI/VEB/VEPA lookup table or VVVL) of 2K entries. Each entry occupies 16 bits for a total of 4 KB of memory. Each entry carries an index of the next table entry in the chain (lower 11 bits) and a TC particular node is associated with (high 3 bits). Each VEB and VSI switching entity might have up to 8 tree nodes allocated for (on node per TC). A *Next* field of the lookup table entry is used to create a linked list of tree nodes allocated for the same VSI or VEB/PA. The index of the lookup table entry equals to the index of the tree node in the nodes table. The switching configuration table entry referred by SEID keeps an index of the first lookup table entry corresponding to one of the tree nodes allocated for respective VSI or VEB/PA. The lookup table allows firmware to identify all tree nodes associated with particular switching components without depending on the ETS-based tree configuration.

A.1.13.4.2 VNet-based Configuration

- VEB/PA nodes.
- Each VEB/PA has only one node instantiated in this configuration. The index of this node is kept in the switch configuration table.
- VSI nodes.
- Each VSI has only one node instantiated in this configuration. The index of this node is kept in the switch configuration table.
- TC/UP nodes.
- TC/UP nodes are allocated per VSI. Firmware does not keep a record of all TC/UP nodes allocated per VSI. Each time firmware needs to modify an ETS configuration of VSI, it reads VSI subtree, consisting of TC and UP nodes allocated for VSI. Firmware does keep a record of UP/TC and each tree node is associated with (4 bits x 2048 = 1 KB). This information is used to match TC/UP indexes with the tree node indexes associated with those TCs/UPs. The order of the TC/UP nodes in the tree is not necessarily matched with UP/TC indexes.

A.1.13.4.3 Shared Bandwidth Limit Table

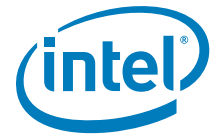
The X710-TM4 supports a limited number of shared bandwidth limit accounts (512). Shared bandwidth limit table entries are allocated on demand and their indexing is not related to the indexing of the nodes table. A bandwidth limit table entry carries an index of the shared bandwidth limit table entry associated with particular node, if any. Firmware can keep a bitmap of 512 bits to keep a map of vacant entries in shared bandwidth limit table.

A.1.13.5 Access Validation

Firmware should restrict the use of AdminQ commands to PF software only.

Firmware also should validate that AdminQ commands that accesses/programs resources owned by a particular PCI function is issued by the PCI function that owns the resource or on behalf of this PCI function.

Firmware should perform validation per switching component or per virtual port. Firmware should use information kept in the corresponding entry of the switching structure representation table. The switching structure representation entry is identified by SEID provided within the AdminQ command.



A.1.13.6VNet-based Configuration Scheme

A.1.13.6.1 VSI Allocation

Allocation and initial configuration of the scheduler configuration tables is done as a part of allocation of the new internal switch VSI. The scheduler does not have a dedicated AdminQ command to allocate a VSI. Allocation and update of the scheduling tables is performed as a part of the internal switch configuration command described in [Section 7.4.9.4.2](#). New VSI is configured with default number of UPs.

VSI allocation does not require an atomic update of the scheduler configuration table and can be done concurrently with the running scheduler. Even when nodes are connected to the scheduler tree, they have no work available and therefore cannot be scheduled. Software associates QPs with newly allocated queue sets only after create VSI call returns.

Perform following scheduling table updates:

- Allocate a VSI node in the node scheduler table.
 - Use available nodes in octet's allocated for the parent (assuming that order is not important for VSIs and it is not unless strict priority arbitration is enabled for VSIs).
 - Do not use nodes octet associated with other switching components.
 - If a node cannot be allocated, allocate a new nodes octet and chain it to octet(s) allocated for the parent switching component.
 - If VSI is a first child of the switching component, update the VSI branch table entry using the parent's branch table entry and CopyAndShift operation. Note that if the switching component is not a first child, then the branch table of the first child among the switching component siblings should be used.
- Configure a VSI node entry.
 - Single bandwidth allocation credit, bandwidth limit and best effort should be disabled by default (no work available).
- Zero out corresponding entries in the bandwidth limit and best effort tables and initialize the corresponding entry in the branch table to list all nodes leading from the VSI node to the tree root node. Firmware can use a preconfigured template in respective tables and CopyEntry operation.
- Allocate the number of UP nodes, as specified AdminQ command.
 - UP nodes owned by different VSIs can share same octet.
 - Update the branch table entry of the first child UP node using a branch table entry of the first child VSI Mode and CopyAndShift operation.
- Configure each of UP nodes:
 - Single bandwidth allocation credit, bandwidth limit and best effort should be disabled by default (no work available).
- For each allocated UP node, zero out the corresponding entries of the bandwidth limit and best effort tables, use preconfigured template and CopyEntry operation.
- Update the VSI node table entry to refer to first and last UP nodes.
- If only one UP was configured for the VSI, firmware can share same tree node for VSI and UP.
- Allocate a queue set for each UP as described in [Appendix A.1.13.3.1](#).
- For each new allocated ready list mapping table entry, configure it with corresponding TC and make it refer to the UP node as a leaf node.
- Store the VSI node index in the switching element entry referred by SEID of the allocated VSI.
- Record queue set indexes and return them to software as QS_Handles in AdminQ completion.



A.1.13.6.2 Configure VSI Bandwidth Limit

Software can enable bandwidth limit for the instantiated VSI. See [Section 7.8.4.5](#) for the AdminQ command definition.

The following steps should be performed to configure bandwidth limit for the VSI:

- Use SEID to locate VSI in the switching structure representation table, [Section 7.4.9.1](#).
- Retrieve the VSI node index.
- Configure the bandwidth limit table entry referred by the node index using WriteEntry operation.
- Enable bandwidth limit in the respective node table entry using WriteField command.

A.1.13.6.3 Configure VSI Bandwidth Allocation

Software can change a default bandwidth allocation for the VSI using AdminQ command described in section [Section 7.8.4.7](#).

Bandwidth allocation is relative. Therefore modification of the bandwidth allocation for the particular VSI should be done with respect to all other VSIs of the same switching component. Software that owns the switching component is required to maintain a map of bandwidth allocations and be responsible for the validity of configuration.

The following steps should be performed to configure bandwidth allocation for the specified VSI.

- Use SEID to locate VSI in the switching structure representation table, [Section 7.4.9.1](#).
- Retrieve the VSI node index.
- Update the credits increment field of the nodes table with a specified number of credits using WriteField command.

A.1.13.6.4 Configure VSI UPs

Software can change the number and UPs configured for the VSI. See [Section 7.8.4.8](#) for AdminQ command definition.

If software has ETS enabled for the VSI, it must disable it prior to changing UP configuration of the VSI; otherwise, firmware fails the UP configuration change. This requirement simplifies firmware flows and management of the scheduler configuration tables. An assumption is that a change in UP configuration would be a rare event and most of the time would be performed prior to ETS being enabled for the VSI in first place.

The following steps should be performed to change the UP configuration of the VSI:

- Use SEID to locate VSI in the switching structure representation table, [Section 7.4.9.1](#).
- Retrieve the VSI node index.
- Identify the children node of the VSI node. Those nodes should be UP nodes.
- Allocate new tree nodes required for new UPs:
 - Firmware can choose reuse already allocated nodes, though from a resource management stand point it seems to be easier to allocate new nodes and deallocate old nodes.
 - Allocating and updating new nodes does not require atomicity and can be performed using the WriteEntry operation.
 - UPs are allowed to share an octet with UPs belonging to different VSIs.
- Order all UP nodes in the tree based on their UP Indexing.
- Program all new allocated nodes with single bandwidth allocation credit, no bandwidth limit, no best effort and no work available.
- For each new allocated node zero-out corresponding entries of the bandwidth limit and best effort tables.



- For the first UP node, update a branch table entry. Firmware can either CopyAndShift entry of the first VSI sibling or can CopyEntry of the previous first UP.
- Update the VSI node table entry to refer to first and last UP nodes.
- For each UP node, allocate an entry in the ready list mapping table or reuse previously allocated for that UP entry and update the leaf node field.
- Firmware must preserve allocation of the queue sets to the UP nodes for the UPs that were kept from the previous configuration.
- Record queue set indexes and return them to software as QS_Handles in AdminQ completion.

A.1.13.6.5 Configure VSI ETS/SLA

Software can enable/disable and modify an ETS/SLA configuration of the specified VSI. See [Section 7.8.4.9](#) for AdminQ command definition.

Software must allocate UPs for the VSI in this command. If software wants to change the UP configuration of the VSI it should use the Configure VSI UP command, described in [Appendix A.1.13.6.4](#).

To enable and modify the ETS/SLA configuration of the VSI, software must specify the complete configuration. Software is not allowed to specify incremental modifications.

Changing the ETS configuration of the VSI might result in changing the tree structure of the active scheduling tree, which requires firmware to use atomic sequences of the updates to reduce impact on scheduler performance.

The following steps should be performed to enable/modify the ETS/SLA configuration of the VSI.

- Use SEID to locate the VSI in the switching structure representation table, [Section 7.4.9.1](#).
- Retrieve the node index.
- Calculate the number of tree levels required for ETS configuration and number of nodes per level.
 - If the ETS configuration has more than one TC enabled and at least one TC has more than one UP mapped to it, then the specified ETS configuration requires two scheduling tree levels.
 - If the TC has only one UP associated with, then the TC should be collapsed and one tree node should be used to represent both the TC and UP.
 - The number of the nodes on the first tree level equals to the number of TCs enabled for ETS.
 - The number of tree nodes on the second tree level equals to the number UPs mapped to each TC, excluding the case when only one UP is mapped to the TC.
- Calculate the total number of TC nodes that need to be added to the tree (all UP nodes are already allocated) and fails the operation if the scheduler configuration table does not have enough free nodes available.

Note: The ready list mapping table should have all entries allocated and associated with the corresponding UP node.

- Allocated UP nodes for all children of the VSI node.
- Identify the UP to map a queue set of the existing configuration. Mapping of the UP to the queue set must remain unchanged. If the UP nodes are migrated to accommodate the scheduling table structure restriction. Note that the respective ready list mapping table entry should be updated to preserve the mapping.
- Allocate new tree nodes required for TCs with more than one UP mapped. TC nodes belonging to a different VSI should share node octets and they can share octets with UP nodes as well. Nodes used for a previous ETS configuration can be either freed or reused depending on the firmware implementation choice.



- For each new allocated TC node, zero-out the corresponding entries of the bandwidth limit and best effort tables. Use a pre-configured template node and CopyEntry operation.
- Update a branch table entry for the first TC node. Use a branch entry of the first VSI child node and CopyAndShift operation.
- Update branch table entries of all first UP nodes using the corresponding first TC branch table entry and CopyAndShift operation.
- Update the VSI node table entry to refer to the first and last TC node allocated for it.
- If total TC credits in the AdminQ command is not zero, allocate the entry in best effort table corresponding to the VSI node and update this entry with the total TC credits. This indicates that the provided configuration is SLA and bandwidth allocation between TCs is best effort.
- If any TC was marked as a strict priority, order allocated TC nodes within the nodes table to have strict priority TCs coming first, ordered accordingly to their priority (TC7-TC0), followed by weighted round robin (WRR) nodes and update WSP to WRR switch the point field of the VSI node table entry with the index of the first WRR node. If all nodes are SWP, then set switching point to 0. If all nodes are WRR, set the switching point to the index of the first child node.
- Allocated UP nodes are identified using first and last child index fields of the VSI node. UP indexes of allocated UP nodes are identified using a lookup table maintained by firmware.
- If any of the TC nodes is a leaf node (such as when the TC has only one UP mapped to it), update a corresponding entry in the ready list mapping table (based on UP) and make the leaf node field refer to the new node. Release previously allocated UP node.
- For all TC nodes that are not leaf nodes, update references to the first and last UP nodes.
- Update bandwidth allocation credits in the nodes table for all TC nodes, including collapsed TC/UP nodes.
- If the TC has a non-zero total UP credits in AdminQ command, allocate the entry in the best effort table corresponding to this TC node and update this entry with the total number of UP credits. This indicates that the provided configuration is SLA and bandwidth allocation between UPs within that TC is best effort.
- If any UP was marked as a strict priority, order allocated UP nodes within the nodes table to have strict priority UPs coming first, ordered accordingly to their priority (UP7-UP0), followed by weighted round robin (WRR) UP nodes and update the WSP to WRR switch point field of the TC node table entry with the index of the first WRR node. If all nodes are SWP, then set the switching point to 0. If all nodes are WRR, set the switching point to the index of the first child node.
- Update bandwidth allocation credits in the nodes table for all UP nodes.

A.1.13.6.6 Configure VSI UP/TC Bandwidth Limit

Software can configure a bandwidth limit to each component of an ETS/SLA configuration. For example, TCs and UPs participating in an ETS/SLA configuration. See [Section 7.8.4.7](#) for AdminQ command definition.

Software must maintain a map of the current ETS configuration and bandwidth limits can be configured only for enabled ETS/SLA configuration components. Attempting to configure bandwidth limit for disabled an ETS/SLA component fails.

Software must use TC and UP handles while referring to the TCs and UPs.

Software cannot provide an incremental update to the ETS bandwidth limit configuration. A bandwidth limit of 0 is used to disable the bandwidth limit.

The following steps should be performed to configure the ETS/SLA bandwidth limit for the VSI.

- UP and TC nodes are contained within the VSI node subtree. Firmware identifies those nodes by reading the corresponding nodes table entries.
 - First children nodes of the VSI using first and last node indexes to identify TC nodes.



- Then for each identified TC node, firmware reads its children nodes, if any available to identify UP nodes.
- Node type and UP/TC index is identified using a lookup table maintained by firmware.
- Software should provide bandwidth limits only for the previously enabled UPs and TCs.
- Update respective entries of the bandwidth limit table using the WriteEntry operation. The bandwidth limit table can be updated using the Immediate command and the nodes table should be updated as an atomic operation.
- Enable bandwidth limit in the respective node table entry using WriteField operation.

A.1.13.6.7 Query VSI Bandwidth Configuration

Software can query for the current bandwidth configuration of the VSI. See [Section 7.8.4.15](#) for the AdminQ command definition.

Since this operation does not change the scheduler table structure it can be done with immediate ReadEntry commands without impacting scheduler performance.

The following steps should be performed to retrieve the current bandwidth configuration of the specified VSI.

- Use SEID to locate the VSI in the switching structure representation table. See [Section 7.4.9.1](#).
- Retrieve the VSI node index.
- Walk the node subtree, identify UP nodes (leaf nodes) and provide a bitmap of valid UPs. A UP index for each identified UP node can be retrieved from the lookup table maintained by firmware.
- Retrieve bandwidth allocation credits from the nodes table entry.
- If the nodes table entry indicates that bandwidth limit is enabled for the node, retrieve the bandwidth limit increment and maximum credits from the corresponding bandwidth limit table.

A.1.13.6.8 Query VSI ETS/SLA Configuration

Software can query for the current ETS/SLA configuration of the specified VSI. See [Section 7.8.4.16](#) for AdminQ command definition.

Since this operation does not change the scheduler table structure it can be done with immediate ReadEntry commands without impacting scheduler performance.

The following steps should be performed to retrieve current ETS/SLA configuration of the specified VSI.

- Use SEID to locate the VSI in the switching structure representation table. See [Section 7.4.9.1](#).
- Retrieve the VSI node index.
- Read the VSI node from nodes table using the ReadEntry operation.
- Check if best effort is enabled and retrieve the TC total credits value from the corresponding best effort table entry.
- Use the first child node index and last child node index to identify the TC nodes.
- Use WSP to WRR switch point to identify TC nodes that are configured to strict priority.
- Read the TC nodes and retrieve the TC bandwidth allocation for each node.
- For each TC node, check whether bandwidth limit is enabled and retrieve the bandwidth limit and bandwidth limit maximum values from the bandwidth limit table.
- For each TC that is not leaf node:
 - Check if best effort enabled for the TC and retrieve UP total credits from the best effort table.
 - Use the first child node Index and a last child node index to identify the UP nodes associated with the TC.



- Use a reference to the ready list mapping table and a TC value in the respective entry to retrieve TC value.
- Use WSP to WRR switch point to identify UP nodes that are configured to strict priority.
- Read the UP nodes and retrieve the UP bandwidth allocation for each node.
- For each UP node, check whether bandwidth limit is enabled and retrieve the bandwidth limit and bandwidth limit maximum values from the bandwidth limit table.

A.1.13.6.9 Switching Component Allocation

Allocation and initial configuration of the scheduler configuration tables is done upon allocation of the new internal switching component. The scheduler does not have a dedicated AdminQ command for that and updating the scheduling tables is performed as a part of the internal switch configuration command described in [Section 7.4.9.4.2](#).

The switching component can be added instead of the already allocated VSI. In that case, the VSI node, including scheduling subtree built on top of it, should be migrated and become a first VSI of the new instantiated switching component. Since this operation involves moving the already configured scheduling tree components, it should be performed using series of atomic operations. In that case a new switching component should take the entry in the table originally occupied by the VSI node and the VSI node should be moved to the new location. In addition to moving VSI node, all descendant nodes that are first siblings, including VSI node itself must have their branch table entries updated.

Perform the following scheduling table updates:

- Allocate a switching component node in the node scheduler table. Due to small number of supported switching components, they should be allocated from the reserved node octets.
- Configure a node entry with a single bandwidth credit, no bandwidth limit, to best effort and no work available.
- Zero out corresponding entries in the bandwidth limit and best effort tables using a preconfigured template node and CopyEntry operations.
- If new added switching component is a first sibling, then initialize the corresponding entry in branch table.
- Store the switching component node index in the switching element entry referred by SEID of the allocated switching component.

A.1.13.6.10 Configure Switching Component Bandwidth Limit

Software can enable a bandwidth limit for the instantiated switching component. See [Section 7.8.4.9](#) for the AdminQ command definition.

The following steps should be performed to configure the bandwidth limit for the switching component:

- Use SEID to locate the switching component in the switching structure representation table. See [Section 7.4.9.1](#).
- Retrieve the node index.
- Configure the bandwidth limit table entries referred by the node index using WriteEntry commands. The bandwidth limit table can be updated using the Immediate command, but the nodes table should be updated as an atomic operation.
- Enable the bandwidth limit in the respective node table entry using the WriteField command.

A.1.13.6.11 Configure Switching Component Bandwidth Allocation

Software can change a default bandwidth allocation for the switching component using this AdminQ command. See section [Section 7.8.4.9](#) for AdminQ command definition.



Bandwidth allocation is relative. Therefore modification of the bandwidth allocation for the particular switching component should be done with respect to all other virtual ports or switching components sharing the same uplink switching component. For example, VEBs and S-channels connected to the same S-comp). Software that owns the switching component is required to maintain a map of bandwidth allocation and is responsible for the validity of the configuration.

The following steps should be performed to configure bandwidth allocation for the specified switching component.

- Use SEID to locate the switching component in the switching structure representation table. See [Section 7.4.9.1](#).
- Retrieve the node index.
- Update the credits increment field of the nodes table with a specified number of credits using the WriteField command.

A.1.13.6.12 Configure Switching Component ETS

This command cannot be used in a VPN-based configuration scheme.

A.1.13.6.13 Configure Switching Component UP/TC Bandwidth Limit

This command cannot be used in a VPN-based configuration scheme.

A.1.13.6.14 Query Switching Component Bandwidth Configuration

Software can query for the current bandwidth configuration of the switching component. See [Section 7.8.4.17](#) for the AdminQ command definition.

Since this operation does not change the scheduler table structure it can be done using immediate ReadEntry commands without impacting scheduler performance.

The following steps should be performed to retrieve the current bandwidth configuration of the specified switching component.

- Use SEID to locate the switching component in the switching structure representation table, [Section 7.4.9.1](#).
- Retrieve the node index.
- Retrieve the bandwidth allocation credits from the nodes table entry using the ReadEntry operation.
- If the nodes table entry indicates that bandwidth limit is enabled for the node, retrieve the bandwidth limit increment and maximum credits from the corresponding bandwidth limit table using the ReadEntry operation.

A.1.13.6.15 Query Switching Component ETS Configuration

This command cannot be used in a VPN-based configuration scheme.

A.1.13.7 ETS-based Configuration Scheme

A.1.13.7.1 VSI Allocation

VSI allocation in an ETS-based environment is more complex process. It involves allocation of up to 8 VSI nodes (one per TC) and adding each node to the respective parent VEB node per TC. All allocated VSI nodes are instantiation of the same VSI per TC.

AdminQ provides firmware with an SEID of the uplink switching component and a bitmap of TCs enabled for VSIs. TCs enabled for VSIs must be a subset of TCs enabled for switching component. In case of S-comp, enabled TCs are defined by the physical port ETS configuration. In case of VEB/PA, TCs are enabled either upon VEB/PA creation or configured using the configure switching component bandwidth allocation or bandwidth limit per the traffic type AQ command.

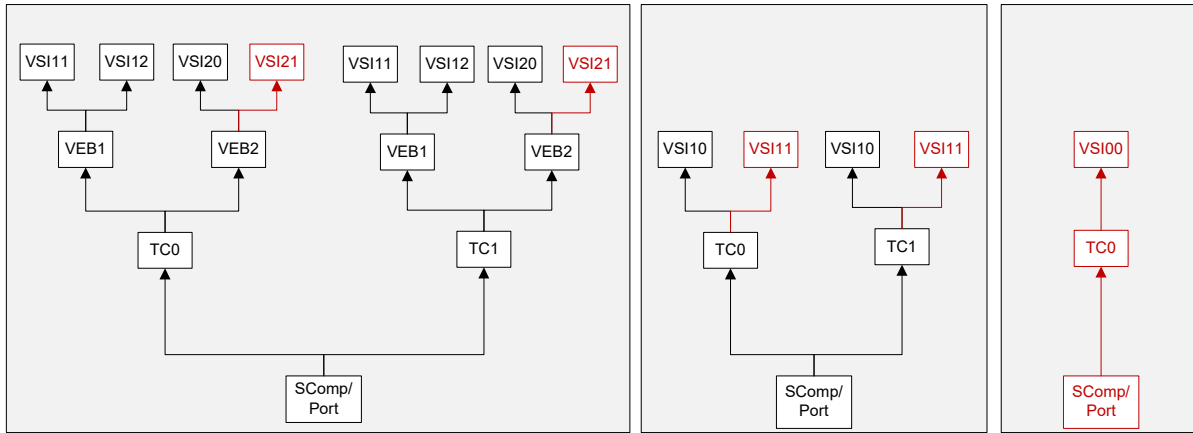


Figure A-15. VSI Allocation Cases

Figure A-15 shows all three VSI allocation cases. New nodes and connections are shown in red. The left side shows adding VSI to VEB/PA. The middle shows adding VSI to S-comp. The right side shows a default VSI allocation for a physical port.

When a port is configured with Multi_Qset mode enabled, each added VSI node comes with a group of eight children and eight Qsets associated with. See Figure A-16.

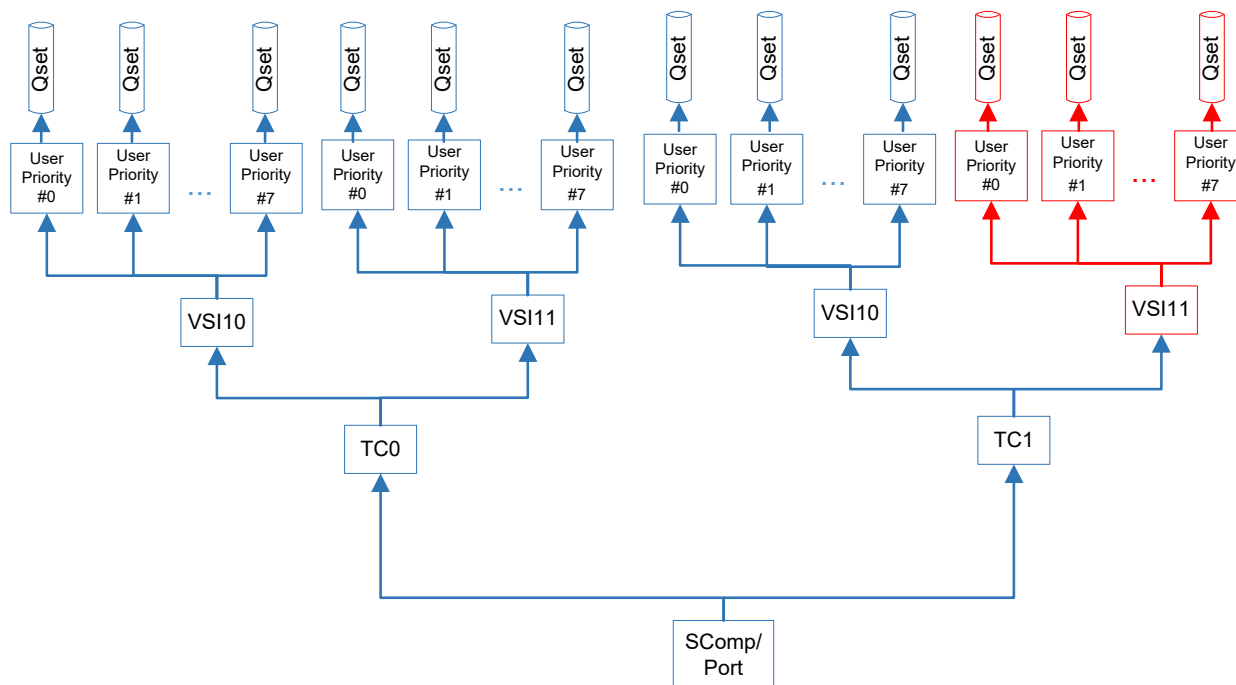


Figure A-16. Topology Structure When Port is Configured with Multi_Qset Mode

The sequence of scheduler table updates is similar to those described for the VNet-based configuration scheme and are omitted here.

In an ETS-based configuration, VSI nodes are usually tree leaf nodes and therefore their allocation results in allocation of the queue sets.

Perform the following scheduling table updates:

- Allocate VSI nodes in the node scheduler table.
 - One node for each enabled TC:
 - If a port is configured with Multi_Qset then:
 - Allocate the octet nodes as children of the VSI node for each one of the enabled TC.
 - Configure the nodes as children of the VSI node.
 - Configure the octet siblings group for strict priority arbitration between the nodes.
 - Use SEID of the parent switching component to identify a VSI/VEB/PA lookup table index kept in the switch configuration table.
 - Identify a parent tree node per TC for each VSI node using a VSI/VEB/PA lookup table. All tree nodes allocated for the parent switching component are chained in this table and each entry carries an TC associated with it. If the parent switching component is S-comp, the VSI should use TC nodes as a parent nodes (middle case in the [Figure A-15](#)).
 - Update the VSI/VEB/PA lookup table entries corresponding to the VSI nodes allocated per TC by creating a linked list.
 - Record a VSI/VEB/PA lookup table index of the first VSI node in the chain in the switching configuration table.
- Update the parent VEB nodes or TC nodes if the VSI is connected directly to S-comp or to a physical port.

- Configure all allocated per TC node entries.
 - For single bandwidth allocation credit, bandwidth limit and best effort should be disabled by default and no work available.
- Zero out corresponding entries in the bandwidth limit and best effort tables and initialize the corresponding entry in the branch table to list all nodes leading from the leaf node to the tree root node. Firmware can use a preconfigured template in the respective tables and CopyEntry operation.
- If Multi_Qset is disabled, then allocate a queue set for each VSI per TC as described in [Appendix A.1.13.3.1](#).
- Else, allocate a queue set for each one of the children of the VSI per TC.
- For each new allocated ready list mapping table entry, configure it with corresponding TC and make it refer to the respective VSI node as a leaf node.
- Record queue set indexes and return them to software as QS_Handles in AdminQ completion.

A.1.13.7.2 Default VSI Allocation

A default VSI is allocated for each physical port in an SFP configuration or for each PF enable for the physical port in MFP mode. Firmware should follow The VSI initialization sequence described in [Appendix A.1.13.6.1](#) assuming TC0 enabled.

A default VSI is configured with the default bandwidth allocation of single credit with no bandwidth limit enabled.

Software needs to use the configure VSI bandwidth allocation per Traffic Type command to enable other TCs for the default VSI.

A.1.13.7.3 Changing DCB Configuration

If DCB is not enabled for the physical port, all VSIs and VEB/PAs are allocated with TC0 enabled. The DCB configuration can be enabled or changed at any point. If DCB is enabled or modified after one or more switching components and VSIs were allocated to the physical port, those components remain configured with prior DCB configuration (including no-DCB) and firmware updates a tree by adding or removing TC nodes accordingly to the new DCB configuration. It is software’s responsibility to configure enabled TCs and the per TC bandwidth configuration of all VSIs and VEB/PAs.

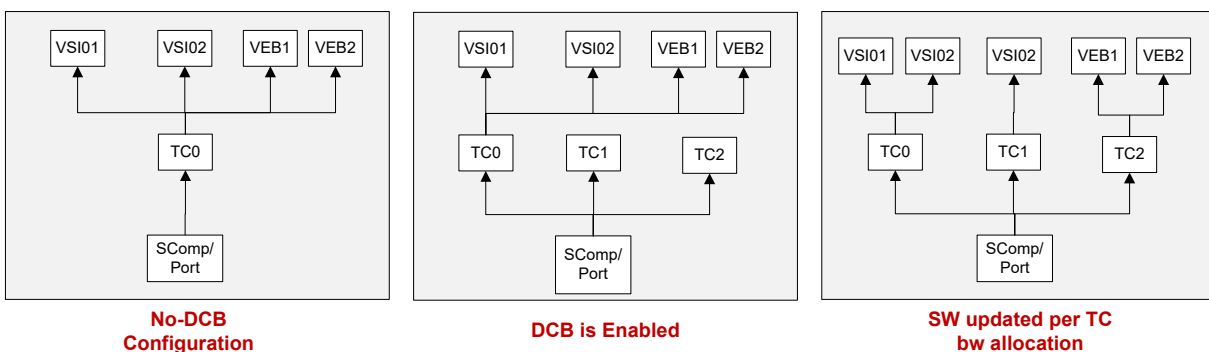


Figure A-17. Enabling DCB



Figure A-17 shows an example of tree transition as a part of DCB enable flow. The left side shows a configuration with multiple VSIs and VEBs allocated for S-comp in the No-DCB configuration. The middle shows new TC nodes added by firmware as a result of DCB enabled for the physical port. The right side shows a tree configuration after software modified TCs enabled and per TC bandwidth allocation for the allocated VSIs and VEB/PAs.

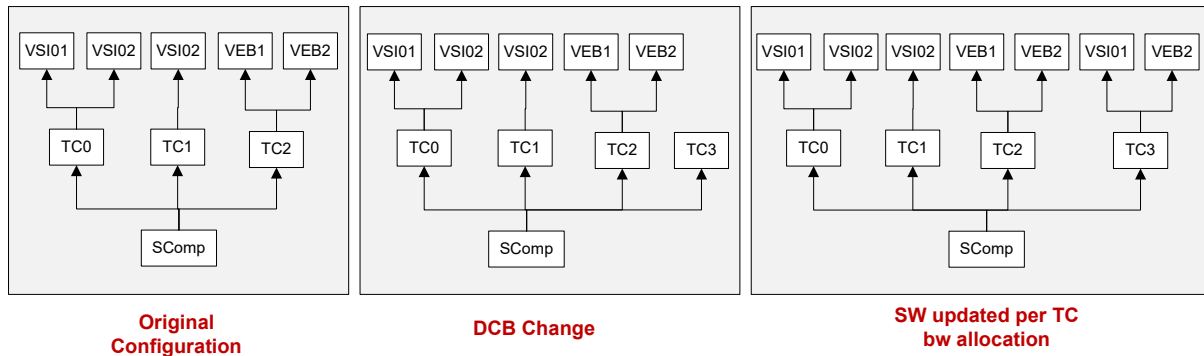


Figure A-18. Increasing the Number of TCs

Figure A-18 shows an example of tree transition as a part of DCB configuration change (number of TCs increased). The left side shows an original configuration with various VSIs and VEBs configured with different TCs. The middle shows a DCB configuration change by adding TC3. The right side shows a new tree configuration after software reconfigured VSIs and VEB/PAs with a new set of TCs.

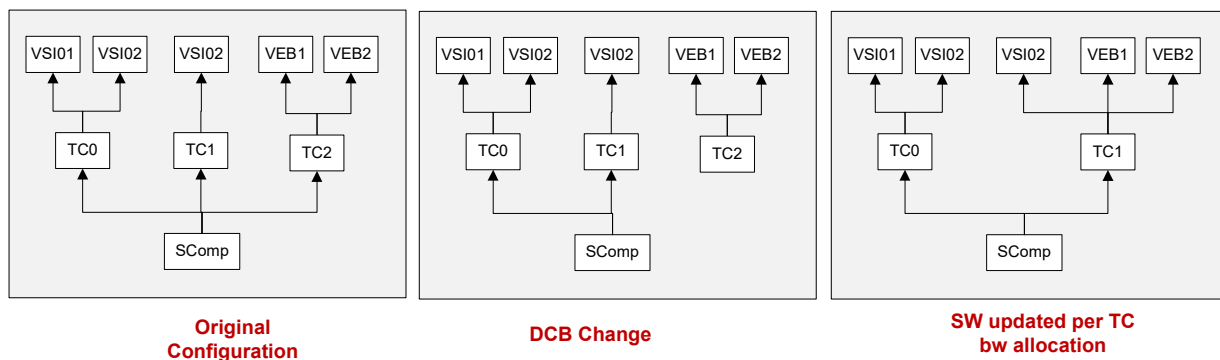


Figure A-19. Decreasing the Number of TCs

Figure A-19 shows an example of tree transition as a part of DCB configuration change (number of TCs decreased). The left side shows an original configuration with various VSIs and VEBs configured with different TCs. The middle shows a DCB configuration change by removing TC2. Note that in this tree, the transmit scheduler can keep scheduling work on unaffected TCs. No work would be scheduled for any queue set associated with TC2. When removing TC2 from the scheduling tree, firmware should request hardware to suspend that TC2 node. The DCB configuration change might not come synchronized with software managing bandwidth and TC for VEBs and VSIs. Firmware should allow



updates of the subtree of the TC disconnected due to the DCB configuration change. Once software is notified about the DCB change by the LLDP firmware agent, software is responsible to adjust TC allocation for VSIs and VEBs to match the ETS configuration of the physical port. The right side shows a new tree configuration after software reconfigured VSIs and VEB/PAs with a new set of TCs.

A.1.13.7.4 Configure VSI Bandwidth Limit

A single VSI system entity can be represented by multiple tree nodes, depending on the number of TCs enabled for the VSI. If the VSI has only one TC enabled, then the process of enabling the bandwidth limit is similar to one described in [Appendix A.1.13.6.2](#).

If the VSI has multiple TCs enabled, then multiple VSI tree nodes are instantiated. In that case, firmware has to instantiate a shared rate limit and configure both private and shared bandwidth limit tables.

The total bandwidth limit configured for VSI should be distributed between the private bandwidth limit accounts (table entries).

Firmware can use one of two options in distributing credits:

- Even distribution among all nodes
- Uneven distribution, relative to the bandwidth allocation credits.

The following steps should be performed to configure the bandwidth limit for the VSIs in case of multiple TCs enabled for the VSIs:

- Use SEID to locate the VSI in the switching structure representation table. See [Section 7.4.9.1](#). This entry refers to the first entry in the VSI/VEB/PA lookup table.
- Walk the linked list of the VSI/VEB/PA lookup table entries and identify all tree nodes allocated for the VSI.
- Allocate and configure the shared bandwidth limit table entry using the WriteEntry operation
- Calculate private credits for each VSI node using one of the algorithms previously described.
- Configure the bandwidth limit table entry referred by the node index using the WriteEntry operation.
- Enable the bandwidth limit in the respective node table entry using the WriteField command.

A.1.13.7.5 Configure VSI Bandwidth Allocation per Traffic Type

Software can configure the VSI bandwidth allocation within each type of traffic. See [Section 7.8.4.8](#) for the AdminQ command definition.

In an ETS-based configuration each VSI does not have its own per-say independent ETS configuration. The VSI ETS configuration is derived from the physical port ETS configuration by hierarchically distributing per-TC/UP bandwidth between VEBs and their VSIs.

A change in the VSI per traffic type bandwidth allocation does not impact the ETS configuration of the physical port. It only impacts the VSI relative bandwidth allocation among other VSIs belonging to the same VEB within a particular TC. Relative bandwidth calculation and credits allocation is the responsibility of software. See [Figure A-20](#).

If the VSI had a bandwidth limit enabled prior to changing number of TCs enabled for the VSI, firmware is responsible to reprogram the private and shared bandwidth limit table entries corresponding to the per TC VSI nodes. See the details that follow.

- Firmware needs to read the bandwidth limit table entries for all TCs that were previously enabled for the VSI.
- Calculate the total amount of credit increments that were allocated to all nodes together.



- Recalculate a new credits increment given the number of TCs currently enabled for the VSI. For example, new credit increment = total number of credits / number of TCs.
- Update the bandwidth limit table entries for each TC enabled for the VSI with a new credits increment.
- No need to reprogram shared bandwidth limit table.

Software can use this AQ command both to change TCs enabled for the VSI and modify the VSI bandwidth allocation within each of enabled TCs.

A change in the bandwidth distribution does not impact the tree structure. A change in TCs enabled for the VSI does lead to a tree structure change. TCs enabled for the VSI must be enabled for the parent switching component. In case of S-comp or physical port, it is a physical port ETS configuration. A change in TCs enabled for the VSI leads to the change in the queue sets enabled (one queue set per TC enabled for the VSI). Queue sets that are not affected by the change (belonging to the TCs that were previously enabled for the VSI) should remain functional during transition. But the bandwidth distribution can be distorted during the transition period.

Software must provide a full configuration of TCs enabled for the VSI. This command can be used both to increase, decrease or change TC allocation for the VSI, and result in addition, removal or moving tree nodes within the scheduling tree.

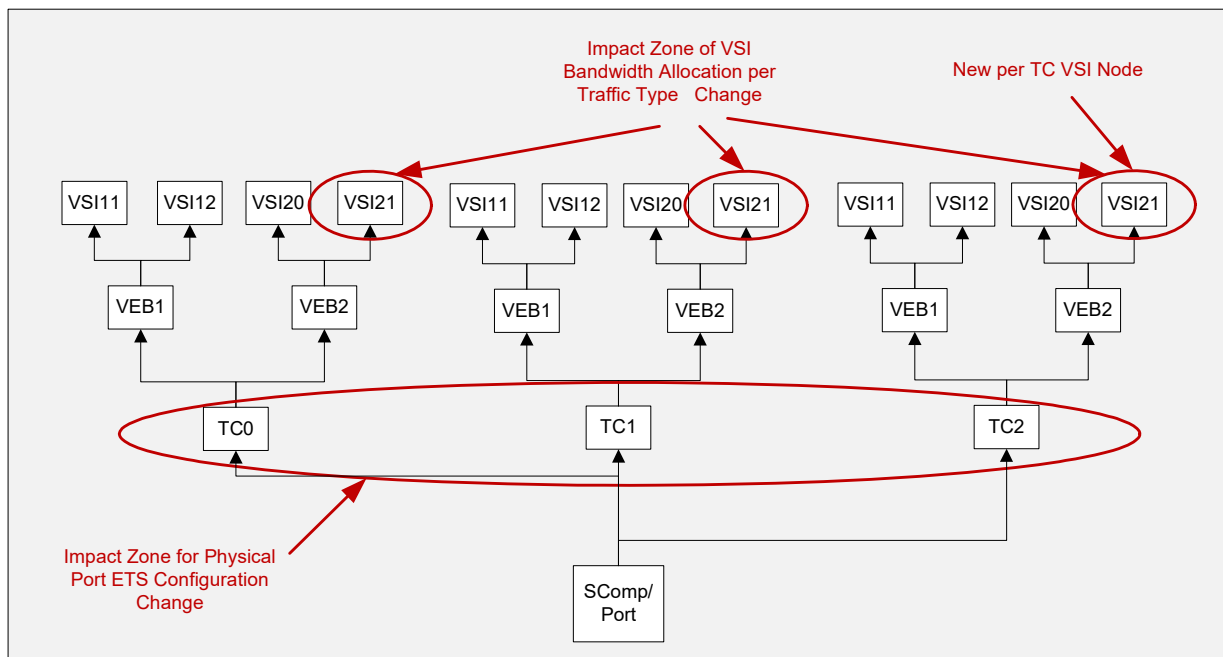


Figure A-20. Example of Per Traffic Type Configuration Change

The following steps should be performed to change TCs enabled for VSIs.

- Use SEID to locate the VSI in the switching structure representation table. See [Section 7.4.9.1](#). This entry refers to the entry in the VSI/VEB/PA lookup table.
- Walk the linked list of the VSI/VEB/PA lookup table entries and identify all tree nodes allocated for the VSI.



- Use SEID of the uplink the switching component to identify an index of the VSI/VEB/PA of the first tree node. This step should be done if the uplink switching component is VEB. Otherwise, firmware should use TC nodes for each TC enabled for the VSI.
- Walk the linked list of the VSI/VEB/PA lookup table entries and identify all tree nodes allocated for the parent switching component.
- Allocate new tree nodes required for new VSI nodes per TC.
 - Firmware can choose to reuse already allocated nodes, though from a resource management stand point it seems to be easier to allocate new nodes and deallocate old nodes.
 - Allocating and updating new nodes does not require atomicity and can be performed using the WriteEntry operation.
- Remove old tree nodes that used to be configured for the VSI and not removed from the list of TCs enabled.
 - Removal of the previously configured nodes requires an atomic update, since those nodes are currently being configured as a part of the scheduler tree.
- Each per TC VSI node is added as a child to the respective VEB node along with other VSI nodes allocated for the VEB. Or, as a child node of the TC node, when the VSI is directly connected to S-comp or to a physical port.
- Program all new allocated nodes with a single bandwidth allocation credit, no bandwidth limit, no best effort and no work available.
- For each new allocated node, zero-out the corresponding entries of the bandwidth limit and best effort tables.
 - If the VSI node is a first sibling, update a branch table entry. Firmware can either CopyAndShift entry of the first VSI sibling or can CopyEntry of the previous first TC.
- Update the respective parent VEB nodes to refer to first and last VSI nodes.
- For each VSI node, allocate an entry in the ready list mapping table or reuse the previously allocated for that VSI entry and update the leaf node field.
- Firmware must preserve allocation of the queue sets to the VSI nodes for the TCs that were kept from the previous configuration.
- Record the queue set indexes and return them to software as QS_Handles in AdminQ completion.

The following steps are performed to change the bandwidth allocation within TCs enabled for VSIs.

- Use SEID to locate the VSI in the switching structure representation table. See [Section 7.4.9.1](#). This entry refers to the entry in the VSI/VEB/PA lookup table.
- Walk the linked list of the VSI/VEB/PA lookup table entries and identify all tree nodes allocated for the VSI.
- Update the bandwidth allocation credits in the nodes table for all VSI UP nodes.

A.1.13.7.6 Configure VSI Bandwidth Limit per Traffic Type

Software can configure the bandwidth limit to the individual VSI node allocated for the particular traffic type. See [Section 7.8.4.7](#) for AdminQ command definition.

Software cannot provide an incremental update to the TC/UP bandwidth limit configuration. A bandwidth limit of 0 is used to disable the bandwidth limit.

In an ETS-based configuration, enabling the bandwidth limit for the VSI on a particular TC effectively means installing a private bandwidth limit on the corresponding VSI TC node.

This AQ command can be used to change TCs enabled for VSIs, similar to configure bandwidth allocation per Traffic Type command described in [Section A.1.13.7.5](#). For the description of adding or removing VSI tree nodes due to a change in TC mapping, see [Section A.1.13.7.5](#).



Software is not allowed to configure both a bandwidth limit for the entire VSI using the AQ command described in [Section A.1.13.7.4](#) and configure bandwidth limit per traffic type. If the VSI already has a bandwidth limit to configure, attempt to configure the bandwidth limit per traffic type fails. The following steps should be performed to configure a TC bandwidth limit for the VSI.

- Use SEID to locate VSI in the switching structure representation table. See [Section 7.4.9.1](#). This entry refers to the entry in the VSI/VEB/PA lookup table.
- Walk the linked list of the VSI/VEB/PA lookup table entries and identify all tree nodes allocated for the VSI.
- Update respective entries of the bandwidth limit table using the WriteEntry operation. The bandwidth limit table can be updated using the Immediate command, but the nodes table should be updated as an atomic operation.
- Enable the bandwidth limit in the respective node table entry using the WriteField operation.

A.1.13.7.7 Query VSI Bandwidth Configuration

Software can query for the current bandwidth configuration of the VSI. See [Section 7.8.4.15](#) for the AdminQ command definition.

Since this operation does not change the scheduler table structure, it can be done with immediate ReadEntry commands without impacting scheduler performance.

The following steps should be performed to retrieve the current bandwidth configuration of the specified VSI.

- Use SEID to locate VSI in the switching structure representation table. See [Section 7.4.9.1](#).
- Use the VSI/VEB/PA lookup table to identify all tree nodes allocated for the VSI and a TC for each node. If VSI nodes have a shared rate limiting account allocate:
 - Read private rate limit accounts for each VSI node in the chain.
 - Sum credits allocated for each node to calculate a VSI bandwidth limit.

A.1.13.7.8 Query VSI Bandwidth Configuration per Traffic Type

Software can query for the current VSI bandwidth configuration per traffic type. See [Section 7.8.4.16](#) for AdminQ command definition.

Since this operation does not change the scheduler table structure it can be done using immediate ReadEntry commands without impacting scheduler performance.

The following steps should be performed to retrieve the VSI bandwidth configuration per traffic type.

- Use SEID to locate VSI in the switching structure representation table. See [Section 7.4.9.1](#).
- Use the VSI/VEB/PA lookup table to identify all tree nodes allocated for the VSI and a TC for each node. Read the VSI node from the nodes table using the ReadEntry operation.
- Update the bandwidth allocation information for each node.
- If VSI nodes have a bandwidth limit enabled, but do not have a same shared rate limit account associated with all nodes, retrieve the bandwidth limit information.

A.1.13.7.9 VSI Deallocation

VSI deallocation can be requested using the switch configuration AQ command (Delete Element). A VSI can be deallocated only if all queue sets associated with a VSI (for all TCs enabled for VSIs) are empty and do not have transmit queues associated with it.

The following steps should be performed to deallocate a VSI.

- Use SEID to locate VSI in the switching structure representation table. See [Section 7.4.9.1](#).



- Use the VSI/VEB/PA lookup table to identify all tree nodes allocated for that VSI.
- Read the VSI nodes from the nodes table using the ReadEntry operation.
- Validate that each VSI node is a leaf node. For an ETS-based scheme this is the only configuration supported.
- Validate that associated queue sets are empty and do not have any transmit queues associated with it.
- Deallocate the VSI nodes.
- There is no need to update work available status of the parent node:
 - Since the queue set associated with the VSI should not have any queue associated with it, the VSI should not have a work available and therefore cannot impact on work available status of the parent node.
 - Even if the VSI node was the only node with work available, not updating work available status for the parent node could lead to one false scheduling in worst case.
- Deallocate queue sets associated with deallocated VSI nodes.

A.1.13.8 Switching Component Configuration

A.1.13.8.1 Switching Component Allocation

Allocation and initial configuration of the scheduler configuration tables is done upon allocation of the new internal switching component. The scheduler does not have a dedicated AdminQ command for that, and updating the scheduling tables is performed as a part of the internal switch configuration command described in [Section 7.4.9.4.2](#).

Adding an S-comp switching component does not change the scheduler tree configuration.

A VEB/PA switching component is usually added as an uplink switching component to the VSI already allocated for the physical port or S-comp, except for the VEB/PA used for internally switched traffic only. To cover all possible configuration changes, firmware should support both.

If VEB/PA is added as an uplink port for existing VSI, a change should be done using a series of atomic operations. In that case, a VSI becomes a first VSI allocated for the VEB and the VEB takes place in the tree previously belonging to the VSI. All TCs that were enabled for the VSI must be enabled for the VEB as well; otherwise, the VEB allocation fails. If additional VSIs need to be added to the VEB, this is done as a separate add VSI operation.

TCs enabled for the VEB must be a subset of TCs enabled for the physical port as a part of the physical port ETS configuration. The VEB might have more TCs enabled than a default VSI configured for the port.

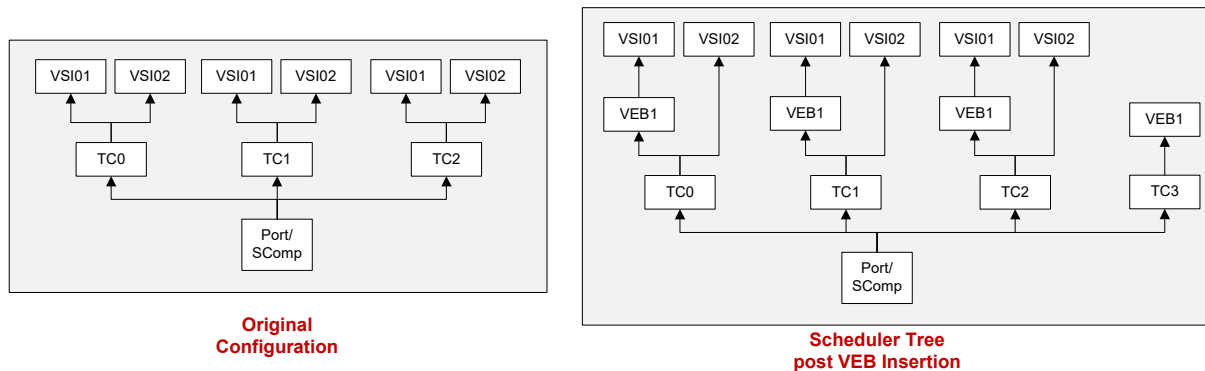


Figure A-21. Example of VEB Insertion

Figure A-21 shows an example of VEB insertion as an uplink of an already allocated VSI (VSI01). The VEB cannot be inserted as an uplink to multiple VSIs, only to one, since multiple VSIs allocated to S-comp has already attached to different S-channels and all VSIs attached to the VEB share the S-channel of VEB. In this example, the VEB is added as an uplink to VSI01, and VSI02 remains directly attached to S-comp. The VEB can be configured with a different number of TCs that are enabled for the VSI. In this example, the VEB has TC0-TC3 enabled, while VSI has only TC0-TC2 enabled. The VEB cannot add TCs that were not enabled for the physical port, so the TC3 tree node should already have been configured as a part of the physical port ETS configuration.

Adding the VEB node does not result in allocation of new queue sets.

Allocation of the VEB/PA switching component involves allocation of multiple tree nodes (one node per TC). Firmware maintains a VSI/VEB/PA lookup table that allows to track all tree nodes allocated to the particular switching component or VSI.

Allocation of tree nodes for the switching components in an ETS-based scheme is similar to allocation of VSI nodes. See [Section A.1.13.7.1](#).

Perform the following scheduling table updates:

- Allocate VEB/PA nodes in the node scheduler table.
 - One node for each enabled TC.
 - Use the UP/TC lookup table or preallocated TC nodes to locate parent TC nodes.
 - Update the VSI/VEB/PA lookup table by chaining tree nodes allocated for the VEB.
 - Record an index of the first VSI/VEB/PA lookup table entry in the switching configuration table allocated for the switching component.
- Update parent TC nodes.
- Configure all allocated per TC VEB/PA node entries.
 - A single bandwidth allocation credit, bandwidth limit and best effort should be disabled by default (no work available).
- Zero out corresponding entries in the bandwidth limit and best effort tables and initialize the corresponding entry in the branch table to list all nodes leading from the VSI node to the tree root node. Firmware can use a preconfigured template in these respective tables and the CopyEntry operation.



- If the VEB was inserted as an uplink port to the VSI, update the corresponding per TC VEB nodes to refer to per TC VSI nodes as a first child nodes.
 - Update the branch table for all per TC VSI nodes.
 - Per TC VEB nodes should inherit the bandwidth configuration of respective per TC VSI nodes. This includes bandwidth allocations and bandwidth limits.\
- If the VEB has more TCs enabled than a default VSI, firmware should allocate the VEB nodes with respective TCs nodes as a parent, with no children nodes and have a default bandwidth allocation configured for those nodes.

A.1.13.8.2 Configure Switching Component Bandwidth Limit

Software can enable a bandwidth limit for the instantiated switching component. See [Section 7.8.4.9](#) for the AdminQ command definition.

Flow of the configuring bandwidth limit for the entire switching component is very similar to the flow of configuring bandwidth limit for the VSI described in [Section A.1.13.7.4](#).

Depending on the number of TCs enabled for the switching component, this might involve a need to allocate and configure the shared bandwidth limit table entry.

For more details on allocation and configuration flow see [Section A.1.13.7.4](#).

A.1.13.8.3 Configure Physical Port ETS

Software can configure ETS of the device's physical port.

Due to small number of TC/UP nodes in an ETS-based configuration, firmware can reserve a set of dedicated tree nodes. Firmware also maintains a lookup table that allows an easy way to locate UP and TC tree nodes based on their UP and TC indexes.

An ETS configuration of the physical port can be changed after the VSI and VEB/PA tree nodes were allocated for the physical port. See [Appendix A.1.13.7.3](#) for more detail.

The following steps are required to configure ETS for the physical port:

- Allocate or use reserved nodes for TCs enabled by the ETS configuration.
- For each new allocated node, zero-out the corresponding entries of the bandwidth limit and best effort tables. Use a pre-configured template node and the CopyEntry operation.
- Update a branch table entry for the first TC node.
- If any TC was marked as a strict priority, order allocated TC nodes within the nodes table to have strict priority TCs coming first, ordered accordingly to their priority (TC7-TC0), followed by weighted round robin (WRR) nodes and update the WSP to WRR switch point field of the VSI node table entry with the index of the first WRR node. If all nodes are SWP, then set the switching point to 0. If all nodes are WRR, set the switching point to the index of the first child node.
- Update the bandwidth allocation credits in the nodes table for all TC nodes.
- If new configuration disabled TC, corresponding TC node must be suspended using the suspend firmware command.
 - Firmware is not responsible to initiate cleanup of the subtree allocated for such TC node.
 - Software must use the configure VSI/VEB/PA bandwidth allocation/limit per traffic type commands to change TC allocation for the affected VSI/VEB/PAs and this drives firmware to cleanup a disconnected subtree.
 - While a suspended TC is disconnected from the scheduling tree, subtree nodes can still be updated due to work available/no work available requests.
 - If the DCB configuration is changed again, a TC node and its subtree can be added back to the scheduling tree using the Resume operation.



- A suspended TC node is released after software released all VSI/VEB/PA nodes corresponding to that TC.

A.1.13.8.4 Configure Switching Component Bandwidth Allocation per Traffic Type

Software can configure the switching component bandwidth allocation per traffic type. See [Section 7.8.4.14](#) for the AdminQ command definition.

This command can be applied to VEB/PA switching components.

Each switching component might have multiple tree nodes allocated, one per TC enabled for the switching component.

This command can be used to change TCs enabled for the VEB. In an ETS-based configuration, the VEB has a separate tree node per TC. TCs enabled for the VEB must be a subset of TCs enabled for the corresponding physical port. If the VEB has VSI allocated, TCs enabled for the VEB must be a superset of TCs enabled for VSIs.

The flow of a scheduler tables update for the VEB/PA bandwidth allocation per traffic type is very similar to one described in [Section A.1.13.7.5](#) for the VSI. Unlike VSIs, the VEB is not associated with queue sets and therefore a change in the TC configuration of the VEB does not result in a change of queue sets allocated.

If software requests to disable a TC previously enabled for the VEB, it must make sure that all VEB VSIs have this TC disabled, prior to making the VEB configuration change.

See [Section A.1.13.7.5](#) for a flow description.

A.1.13.8.5 Configure Switching Component Bandwidth Limit per Traffic Type

Software can configure a bandwidth limit to an individual switching component node allocated for the particular traffic type. See [Section 7.8.4.13](#) for the AdminQ command definition.

Software cannot provide an incremental update to the TC bandwidth limit configuration. A bandwidth limit of 0 is used to disable the bandwidth limit. If the switching component is connected directly to the physical port (S-comp), then the bandwidth limit would need to be configured for the particular TC tree node.

Software can use this AQ command to change the TC enabled for the VEB/PA. TCs enabled for the VEB must be a subset of the TCs enabled for the corresponding physical port. If the VEB has a VSI allocated, the TCs enabled for the VEB must be a superset of the TCs enabled for VSIs.

The flow of the scheduler tables update for the VEB/PA bandwidth limit per traffic type is very similar to the one described in [Section A.1.13.7.6](#) for the VSI. Unlike VSIs, the VEB is not associated with a queue set and therefore a change in the TC configuration of the VEB does not result in a change of queue sets allocated.

If software requests to disable the TC previously enabled for the VEB, it must make sure that all VEB VSIs have this TC disabled, prior to making a VEB configuration change.

See [Section A.1.13.7.6](#) for a flow description.

A.1.13.8.6 Query Switching Component Bandwidth Configuration

Software can query for the current bandwidth configuration of the switching component. See [Section 7.8.4.17](#) for the AdminQ command definition.

Since this operation does not change the scheduler table structure it can be done using Immediate ReadEntry commands without impacting scheduler performance.

The flow of retrieving the VEB/PA configuration is similar to the flow described in [Section A.1.13.7.7](#) for the VSI.



A.1.13.8.7 Query Physical Port ETS Configuration

Software can query for the current ETS configuration of the specified switching component. See [Section 7.8.4.18](#) for the AdminQ command definition.

Since this operation does not change the scheduler table structure it can be done using Immediate ReadEntry commands without impacting scheduler performance.

This command is only valid for the switching component connected directly to the physical port. Otherwise, a request should be rejected and error returned.

The following steps should be performed to gather the physical port ETS configuration

- Firmware should use the TC/UP lookup table to gather information about TCs.
- For each TC node:
 - Read the TC node entry in the nodes table.
 - Use the X710-TM4 registers to recover UP-to-TC mapping.
 - Gather per TC bandwidth allocation credits.
 - If the TC node had a bandwidth limit enabled, read the corresponding entry of the bandwidth limit table.
- For each UP node:
 - Provide the gathered data once the AdminQ command completes.

A.1.13.8.8 Query Switching Component Bandwidth Configuration per Traffic Type

Software can query for the switching component bandwidth configuration per traffic type. See [Section 7.8.4.19](#) for the AdminQ command definition.

Since this operation does not change the scheduler table structure it can be done using Immediate ReadEntry commands without impacting scheduler performance.

The flow of retrieving the VEB/PA configuration is similar to the flow described in [Section A.1.13.7.8](#) for the VSI.

Since the VEB is not associated with queue sets, this step in the flow described in [Section A.1.13.7.8](#) should be omitted.

A.1.14 Scheduler FLR/VFLR Support

During a FLR or VFLR event, hardware scans the ready list mapping table, identifies ready lists owned by the PCI function, suspends all RLM entries by setting a *Suspend* bit and schedules a special pipe-cleaner token. This token travels through LAN pipelines. Completion of hardware pipeline cleanup is reported in TBD register. Firmware uses this register as an indication that it can perform scheduler cleanup for the VF/PF. Token is scheduled one for each pipeline, regardless number of ready lists affected.

Firmware cannot assume that hardware actually ran no work available for each suspended ready lists and therefore not allowed to clear the *Suspend* bit in the RLM table entry without reassigning it to a different PCI function. During FLR/VFLR flow, hardware only sets a *Suspend* bit in the ready list and runs the no work available flow only if the suspended ready list is being scheduled, as a part of the mis-scheduling processing.

Hardware guaranties that upon completion of the hardware cleanup no new work be generated for the FLR'ed or VFLR'ed functions.



Firmware is responsible for the cleanup and deallocation of the scheduler resources upon completion of hardware pipe cleanup flow.

A.1.14.1 Standard Scheduler Configuration

In standard configuration scheduler, follow the configuration of the internal switching fabric. It assumes the following rules of the switching components allocation to PCI functions.

- S-comp is either owned by the physical function or by firmware in case it is shared by multiple PFs (MFP model). A single PF might own at most one S-comp.
- VEB/VEPA is owned by single PF. A single PF might own one or more VEBs.
- VSI is owned by a PF or VF. A single PF or VF might own one or more VSIs.

The following steps should be performed by firmware to properly modify the scheduler resource allocation for a PF FLR event.

- Scan the switching structure representation table. See [Section 7.4.9.1](#).
- For each switching component owned by a PF:
 - Retrieve the node index
 - All queue sets owned by a PF and respective VFs should be suspended now and not processed by the scheduler.
 - All VSIs and switching components owned by a VF/PF should not have work-available and therefore should be skipped by the scheduler during the scheduling process.
- Walk suspended subtrees, cleanup nodes and release ready list mapping table entries.
 - Clean does not require atomicity, since it is performed on the nodes that are not processed by the scheduler.

The following steps should be performed by firmware to properly modify the scheduler resource allocation for a VF FLR event.

- Scan the switching structure representation table. See [Section 7.4.9.1](#).
- For each virtual port owned by a VF:
 - Retrieve the node Index
 - All queue sets owned by a VF should be suspended now and not processed by the scheduler.
 - All VSIs owned by a VF should not have work-available and therefore should be skipped by the scheduling during the scheduling process.
- Walk suspended subtrees, cleanup nodes and release ready list mapping table entries.
 - Clean does not require atomicity, since it is performed on the nodes that are not processed by the scheduler.



NOTE: *This page intentionally left blank.*