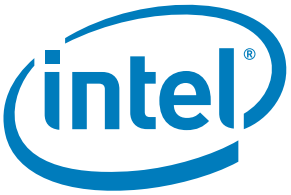


Intel® Ethernet Controller I225

Software User Manual

Intel Confidential

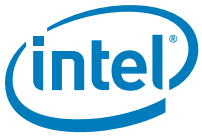
Revision: *1.3 (based on internal EAS rev 0.99)*

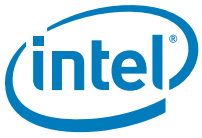
Publish Date: *17-Aug-2020*

Intel Confidential

Revision History

Rev	Date	Authors	Comments
1.3	3-Aug-2020	Avi Shalev	Remove NVM offset 0x20 bit 13 "PHY_in_LAN_dis". Changed to "Reserved". Removed ULP entry by BIOS (5.7.1.1) Updated ULP entry by OS (5.7.1.2) with LAN_DISABLE assertion by BIOS ULP Exit (5.7): Add section 5.7.2.1 for ULP exit by host and 5.7.2.2 for exit by CSME Power supply (2.7) Added Leakage protection section. Added PM capabilities host slave command for magic packet and ULP in D3cold Added external loopback configuration (section 3.6.5) Added B.3 device IDs
1.2	17-Nov-2019	Amir Zinaty	Define which modules are write protected in the NVM and document it in the "NVM Module Names" table Add section "Registers Initialization by Software" that lists those registers that the software driver should revert the values of some fields Changes in the ULP Flow Add the "Set Filter Indirect Table Select" command
1.1	2-July-2019	Amir Zinaty	Change pin names according to the Datasheet Fix the introduction section indication that LTR is supported. Added a note that HDX is supported but SW support might not be continued. Added ULP to the summery table Added PTM to the summary table Some changes in the NVM and the Register chapters Update the flows entering the Dx states and the ULP state Document the PTM reporting endianess
1.0	28-Mar-2019		Initial Revision





1.0 Introduction

The Intel® Foxville 2.5 Gbps Ethernet Controller (Foxville) is a single port, compact, low power component that supports 2.5 GbE designs. Foxville offers a fully-integrated GbE Media Access Control (MAC) and Physical Layer (PHY) port. Foxville supports PCI Express* Gen 2 x one lane [PCIe Gen2 v3.1].

Foxville enables 10M/100M/1000M/2.5G BASE-T implementations using an integrated PHY. It can be used for client and server system, in an add-on NIC or LAN on Motherboard (LOM) designs. It can also be used in various embedded applications.

1.1 Feature Summary

External Interfaces provided:

- PCIe Gen2 v3.1 called PCIe in this document.
- MDI (Copper) standard IEEE 802.3 Ethernet interface for 2.5GBASE-T, 1000BASE-T, 100BASE-TX, and 10BASE-Te applications (802.3, 802.3u, 802.3ab and 802.3bz)
- NC-SI over MCTP over PCI-E or SMBus for Manageability connection to BMC
- IEEE 1149.6 JTAG

Performance Enhancements:

- Intel® I/O Acceleration Technology v3.0 supported:
  - Stateless offloads (Header split, RSS)
- UDP, TCP and IP Checksum offload
- TCP Transmit Segmentation Offload (TSO)
- SCTP receive and transmit integrity offload
- Queues: 4 TX and 4 RX queues

Power saving features:

- Advanced Configuration and Power Interface (ACPI) power management states and wake-up capability
- Advanced Power Management (APM) wake-up functionality
- PCIe v2.1 LTR (ECN - Latency Tolerance Reporting) for improved system power management.
- PCIe v3.1 L1 sub-states
- DMA Coalescing for improved system power management.
- EEE (IEEE802.3az) for reduced power consumption during low link utilization periods.
- Ultra Low Power consumption when the Ethernet link is disconnected.

Time Sensitive Networking support - TSN:

- IEEE 1588 Precision Time Protocol support

- Per-packet timestamp
- 802.1AS-Rev 1588 profile for TSN and dual 1588 timers
- 802.1Qav, 802.1Qbv - Credit based shaping, Basic scheduling and time aware shaper for TSN related applications
- 802.1Qbu and 802.3br - Frame Preemptions and Interspersing Express Traffic
- Note that TSN is supported only in FDX network topology

Total Cost Of Ownership (TCO):

- IPMI BMC pass-thru
- Internal management controller to OS and OS to management controller traffic support
- Interface to external Itel management processor - CSME
- Circuit Breaker features mitigating propagation of worms and viruses

Additional product details:

- 7mm x 7mm VGFN-567 package
- Memory Parity or ECC protection
- Estimated power consumption of 800 mW

1.2 Scope

This document provides the external architecture (including device operation, pin descriptions, register definitions, etc.) for Foxville.

This document is a reference for design groups, architecture validation, firmware development, software device driver developers, board designers, test engineers, and others who may need specific technical or programming information.

1.3 Terminology and Acronyms

Table 1-1. Glossary

Definition	Meaning
2.5GBASE-T	2.5GBASE-T is the specification for 2.5 Gb/s Ethernet over category 5e twisted pair cables as defined in IEEE 802.3 bz.
1000BASE-T	1000BASE-T is the specification for 1 Gb/s Ethernet over category 5e twisted pair cables as defined in IEEE 802.3 clause 40.
AEN	Asynchronous Event Notification
AVB	Audio Video Bridging (is replaced with TSN)
b/w	Bandwidth.
BIOS	Basic Input/Output System.
BMC	Baseboard Management Controller - often used interchangeably with Manageability Controller (MC). Foxville supports only the Intel management controller - ME.
BT	Bit Time.
CRC	Cyclic redundancy check
DDOFF	Dynamic Device Off
DFT	Design for Testability.
DQ	Descriptor Queue.

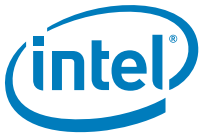


Table 1-1. Glossary (Continued)

Definition	Meaning
DMTF	Distributed Management Task Force standard body.
DW	Double word (4 bytes).
EAS	External Architecture Specification.
EEE	Energy Efficient Ethernet - IEEE802.3az standard
EEPROM	Electrically Erasable Programmable Memory. A non-volatile memory located on the LAN controller that is directly accessible from the host.
EOP	End of Packet.
FC	Flow Control.
FCS	Frame Check Sequence.
Firmware (FW)	Embedded code on the LAN controller that is responsible for the implementation of the NC-SI protocol and pass through functionality.
Host Interface	RAM on the LAN controller that is shared between the firmware and the host. RAM is used to pass commands from the host to firmware and responses from the firmware to the host.
IPG	Inter Packet Gap.
IPMI	Intelligent Platform Management Interface specification
LAN (auxiliary Power-Up)	The event of connecting the LAN controller to a power source (occurs even before system power-up).
LLDP	Link Layer Discovery Protocol defined in IEEE802.1AB used by IEEE802.3az (EEE) for system wake time negotiation.
LOM	LAN on Motherboard.
LPI	Low Power Idle - Low power state of Ethernet link as defined in IEEE802.3az.
LSO	Large Send Offload.
LTR	Latency Tolerance Reporting (PCIe protocol)
iSVR	Integrated Switching Voltage Regulator
MAC	Media Access Control.
MC	Management Controller
MCTP	DMTF Management Component Transport Protocol (MCTP) specification. A transport protocol to allow communication between a management controller and controlled device over various transports.
MDIO	Management Data Input/Output Interface over MDC/MDIO lines.
ME	Intel management engine embedded in the Intel chipset
MIFS/MIPG	Minimum Inter Frame Spacing/Minimum Inter Packet Gap.
MMW	Maximum Memory Window.
MSS	Maximum Segment Size. Largest amount of data, in a packet (without headers) that can be transmitted. Specified in Bytes.
MPS	Maximum Payload Size in PCIe specification.
MTU	Maximum Transmit Unit. Largest packet size (headers and data) that can be transmitted. Specified in Bytes.
NC	Network Controller.
NC-SI	Network Controller Sideband Interface DMTF Specification
NIC	Network Interface Controller.
OBFF	Optimized Buffer Flush/Fill (PCIe protocol).
PCS	Physical Coding Sub layer.
PHY	Physical Layer Device.
PMA	Physical Medium Attachment.
PMD	Physical Medium Dependent.

Table 1-1. Glossary (Continued)

Definition	Meaning
SA	Source Address.
SDP	Software Defined Pins.
SFD	Start Frame Delimiter.
SMBus	System Management Bus. A bus that carries various manageability components, including the LAN controller, BIOS, sensors and remote-control devices.
SVR	Switching Voltage Regulator
TCO	Total Cost of Ownership (TCO) System Management.
TLP	Transaction Layer Packet in the PCI Express specification.
TSN	Time Sensitive Networking set of specifications
TSO	Transmit Segmentation offload - A mode in which a large TCP/UDP I/O is handled to the device and the device segments it to L2 packets according to the requested MSS.
VLAN	Virtual LAN
VPD	Vital Product Data (PCI protocol).

1.3.1 External Specification and Documents

Foxville implements features from the following specifications.

1.3.1.1 Network Interface Documents

1. IEEE standard 802.3, 2006 Edition (Ethernet). Incorporates various IEEE Standards previously published separately. Institute of Electrical and Electronic Engineers (IEEE).
2. IEEE standard 1149.1, 2001 Edition (JTAG). Institute of Electrical and Electronics Engineers (IEEE)
3. IEEE Std 1149.6-2003, IEEE Standard for Boundary-Scan Testing of Advanced Digital Networks, IEEE, 2003.
4. IEEE standard 802.1Q for VLAN
5. PICMG3.1 Ethernet/Fibre Channel Over PICMG 3.0 Draft Specification, January 14, 2003, Version D1.0
6. Serial-GMII Specification, Cisco Systems document ENG-46158, Revision 1.7
7. IEEE Std 802.3ap-2007
8. IEEE 1588™ Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, November 8 2002
9. IEEE 802.1AS Timing and Synchronization for Time- Sensitive Applications in Bridged Local Area Networks 2011
10. IEEE 802.1AS-Rev Timing and Synchronization for Time Sensitive Applications Draft D5.0 June 12, 2017.
11. IEEE P802.1Qbv Bridges and Bridged Networks - Amendment: Enhancements for Scheduled Traffic. Draft D2.3 April 3, 2015
12. IEEE 802.1Qbu Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks—Amendment: Frame Preemption Draft D2-0 December 1, 2014
13. IEEE 802.3br IEEE Standard for Ethernet - Amendment 5: Specification and Management Parameters for Interspersing Express Traffic, Approved 30 June 2016
14. IEEE 802.1BF Ethernet Support for the IEEE P802.1AS Time Synchronization Protocol Task Force
15. IEEE 802.3az Energy Efficient Ethernet Draft 1.4, May 2009
16. 802.1BA - Audio Video Bridging (AVB) Systems

17. 802.1Qav - Forwarding and Queuing Enhancements for Time-Sensitive Streams

1.3.1.2 Host Interface Documents

18. PCI Express Specification: Gen 2, version 3.1
19. PCI Bus Power Management Interface Specification, Rev. 1.2, March 2004
20. Advanced Configuration and Power Interface Specification, Rev 2.0b, October 2002

1.3.1.3 Networking Protocol Documents

1. IPv4 specification (RFC 791)
2. IPv6 specification (RFC 2460)
3. TCP/UDP specification (RFC 793/768)
4. SCTP specification (RFC 2960)
5. ARP specification (RFC 826)
6. Neighbor Discovery for IPv6 (RFC 2461)
7. EUI-64 specification, http://standards.ieee.org/regauth/oui/tutorials/EUI64.html.

1.3.1.4 Manageability Documents

1. DMTF Network Controller Sideband Interface (NC-SI) Specification rev 1.0.0, May 2009
2. DMTF NC-SI over MCTP Binding Specification, rev 1.0.0, 8/22/2013.
3. Management Component Transport Protocol (MCTP) Base Specification, rev 1.1.0, 4/22/2010 which specifies the Management Component Transport Protocol (MCTP)
4. Management Component Transport Protocol (MCTP) SMBus/I2C Transport Binding Specification, rev 1.0.0, 7/28/2009 which describes the binding of MCTP over SMBus.
5. Management Component Transport Protocol (MCTP) PCIe VDM Transport Binding Specification, rev 1.0.1, 12/11/2009 which describes the binding of MCTP over PCI Express.
6. Management Component Transport Protocol (MCTP) IDs and Codes, rev 1.1.0, 11/3/2009 which describes constants used by MCTP specs.
7. Document: dmtf.org/sites/default/files/standards/documents/DSP0236_1.1.0.pdf
8. System Management Bus (SMBus) Specification, SBS Implementers Forum, Ver. 2.0, August 2000

1.3.1.5 Proxy Documents

1. proxZZZy™ for sleeping hosts, February 2010 (ECMA-393)
2. mDNS Offload - Draft 1.0, May 2010

1.4 Product Overview

Foxville is a derivative of previous generations of Intel Gigabit Ethernet controller designs. Many features of its predecessors remain intact, some of them have been removed or modified and new features were introduced.

Foxville supports an internal 2.5G-T PHYs that can be used to implement a single port LAN interface in a Thunderbolt Docking, as a NIC or a LOM design. Targeting mainly the client market.

On top of it, Foxville supports the Intel vPro and Time Sensitive Networking. As such, Foxville can be used also in AVB applications as well as industrial applications that require TSN capabilities.

1.4.1 Time Sensitive Networking Support

Foxville supports TSN standards that are aimed to support time sensitive traffic for AVB and other industrial applications. Foxville provides superset capabilities of its predecessor i210. On top of the basic 1588 and scheduling supported in i210, Foxville supports also the 802.1AS-Rev (dual 1588 timers), 802.1Qbv (time Aware Shaper), 802.1Qbu and 802.3br (Frame Preemption and Interspersing Express Traffic). All of the above is supported only when operating in FDX mode.

1.5 External Interface

1.5.1 PCIe Interface

The PCIe Gen2 v3.1 Interface is used by Foxville as a host interface. The interface only supports the PCIe Gen2 v3.1 rate and is configured to x1. The maximum aggregated raw bandwidth for a typical PCIe Gen2 v3.1 configuration is 4 Gb/s in each direction. Refer to Section 2.3 for a full pin description. The timing characteristics of this interface are defined in the PCI Express Card Electromechanical Specification rev 2.0 and in the PCIe Gen2 v3.1 specification.

1.5.2 Network Interfaces

Foxville provide a single port MDI (copper) for standard IEEE 802.3 Ethernet interface at 2.5GBase-T, 1000BASE-T, 100BASE-TX, and 10BASE-Te applications (802.3, 802.3u, 802.3ab and 802.3bz).

Refer to Section 1.5.2 for additional information.

1.5.3 Serial Flash Interface

Foxville provides an external SPI serial interface to a Flash for storing product configuration information and a boot ROM device. Foxville supports serial Flash devices with up to 64 Mb (8 MB) of memory. The size of the Flash used by Foxville can be configured by the Flash itself. Refer to Section 2.4 for full pin description.

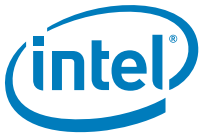
1.5.4 SMBus Interface

SMBus is an optional interface. It is required only for vPro capabilities for pass-through and/or configuration traffic between a Manageability Controller (MC) and Foxville.

Foxville's SMBus interface can be configured to support both slow and fast timing modes (up to 1Mb/s). Refer to Section 2.4 for full pin description and section ***TBD*** for timing characteristics of this interface.

1.5.5 Internal PHY

Foxville implements an internal (integrated) PHY that supports the following link speeds over base T copper link: 10Mb/s; 100Mb/s; 1Gb/s and 2.5Gb/s. Foxville supports the auto-negotiation protocol selecting the highest common denominator speed supported by the link partner.



1.5.6 Foxville SKU Options

Foxville is offered in one of the following SKU options:

Table 1-2. Foxville SKU Options

SKU Name	Description	Main Supported Features			
		2.5G	vPro	TSN	Industial Temp.
Foxville LM	Client 2.5G LAN - Corporate + vPro	V	V	V	-
Foxville V	Client 2.5G LAN - Consumer	V	-	-	-
Foxville IT	Client 2.5G - Industrial Temp	V	V	V	V
Foxville Def.	Client Foxville with Empty Flash Image	-	-	-	-

Table 1-3. Foxville Device/Rev IDs (B-Step)

SKU Name	PCI		Branding Name
	Device ID	Rev ID	
Foxville LM	0x15F2	0x1	Intel(R) Ethernet Controller I225-LM
		0x2	Intel(R) Ethernet Controller (2) I225-LM
		0x3	Intel(R) Ethernet Controller (3) I225-LM
Foxville V	0x15F3	0x1	Intel(R) Ethernet Controller I225-V
		0x2	Intel(R) Ethernet Controller (2) I225-V
		0x3	Intel(R) Ethernet Controller (3) I225-V
Foxville IT	0x0D9F	0x2	Intel(R) Ethernet Controller (2) I225-IT
		0x3	Intel(R) Ethernet Controller (3) I225-IT
Foxville Def.	0x15FD	0x1	Client Foxville with Empty Flash Image

1.5.7 Software-Definable Pins (SDP) Interface (General-Purpose I/O)

Foxville has four software-defined pins (SDP pins) that can be used for IEEE1588 auxiliary device connections, enable/disable of the device, and for other miscellaneous hardware or software-control purposes. These pins can be individually configurable to act as either standard inputs, General-Purpose Interrupt (GPI) inputs or output pins. The default direction of each pin is configurable via the "Software Defined Pins Control" word in the NVM (see also CTRL and CTRL_EXT registers), as well as the default value of all pins configured as outputs. Information on SDP usage can be found in Section 3.4 and Section 7.5.1.3.6. Refer to Section 2.5 for pin description of this interface.

1.5.8 LED Interface

Foxville implements output drivers intended for driving external LED circuits. Each of the three LED outputs can be individually configured to select the particular event, state, or activity, which is indicated on that output. In addition, each LED can be individually configured for output polarity as well as for blinking versus a non-blinking (steady-state) indication.

The configuration for LED outputs is specified via the LEDCTL register. Furthermore, the hardware-default configuration for all LED outputs can be specified via NVM Flash fields (refer to Section 6.1.1.28 and Section 6.1.1.31), thereby supporting LED displays configurable to a particular OEM preference.

Refer to Section 2.0 for full pin description of this interface.

Refer to Section 3.4.3 for more detailed description of LED behavior.

1.5.9 Feature Summary

The Table 1-4 through Table 1-4 summarize the feature set provided by Foxville and compares them to other recent Intel GbE LAN controllers.

Table 1-4. General Features

Feature	Foxville	Springville	Pearsonville	Powerville	82574
Serial Flash interface	Y	Y[1]	N	Y	Y
Integrated NVM (iNVM)	N	Y[2]	Y	N	N
4-wire SPI EEPROM interface	N	N	N	Y	Y
Configurable LED operation for software or OEM custom-tailoring of LED displays	Y	Y	Y	Y	Y
Protected Flash space for private configuration	Y	Y[1]	N	Y	Y
Device disable capability	Y	Y	Y	Y	Y
Package size (mm x mm)	7x7	9x9	9x9	17x17/25x25	9x9
Embedded thermal sensor	N	N	N	Y	N
Embedded thermal diode	N	N	N	Y	N
Watchdog timer	Y	Y	Y	Y	Y
Boundary-Scan IEEE 1149.1	Y	Y	Y	Y	Y
Boundary-Scan IEEE 1149.6	Y	Y	Y	Y	N
Industrial temp (special SKU)	Y	Y	N	N	Y

1. Not applicable in Flash-less Springville operation.
2. Flash-less Springville operation is supported (with no support for manageability related functionalities). Refer to the note that describes the limitation in Section 1.5.3.

Table 1-5. Network Features

Feature	Foxville	Springville	Pearsonville	Powerville	82574
2.5 Gb/s BASE-T	Y	N	N	N	N
10BASE-Te, 100BASE-T, 1000BASE-T	Y	Y	Y	Y	Y
Integrated BASE-T PHY	Y	N			
External PHY control I/F MDC/MDIO 2-wire I/F	N	Y	N	Shared or per function	N

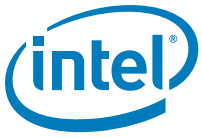


Table 1-5. Network Features (Continued)

Feature	Foxville	Springville	Pearsonville	Powerville	82574
MDI Lane Swap	Y	N	N	Y	N
SerDes interface for external PHY connection or system interconnect	N	Y	N	4 ports	N
1000BASE-KX interface for blade server backplane connections	N	Y	N	Y	N
802.3ap Backplane Auto-negotiation	N	N	N	N	N
SGMII interface for external 1000BASE-T PHY connection	N	1 port	N	4 ports	N
SerDes support of non-auto-negotiation partner	N	Y	N	Y	N
SerDes signal detect	N	Y	N	Y	N
Half duplex at 10/100 Mb/s operation and full duplex operation at all supported speeds	Y [1]	Y	Y	Y	Y
Jumbo frames supported	Y (w/no TSN)	Y (w/no AVB)	Y	Y	Y
Size of jumbo frames supported	9.5 KB	9.5 KB	9.5 KB	9.5 KB	9018B
Flow control support: send/receive PAUSE frames and receive FIFO thresholds	Y	Y	Y	Y	Y
Statistics for management and RMON	Y	Y	Y	Y	Y
802.1q VLAN support	Y	Y	Y	Y	Y
802.3az EEE support	Y	Y	Y	Y	N

1. Half duplex is supported at the silicon level. However, its support by the software driver might not be continued.

Table 1-6. Host Interface Features

Feature	Foxville	Springville	Pearsonville	Powerville	82574
PCIe revision	3.1	2.1	2.1	2.1 (5 Gb/s or 2.5 Gb/s)	1.1
PCIe physical layer	Gen 2	Gen 1	Gen 1	Gen 2	Gen 1
Bus width	x1	x1	x1	x1, x2, x4	x1
64-bit address support for systems using more than 4 GB of physical memory	Y	Y	Y	Y	Y
Outstanding requests for Tx buffers per port	6	6	6	24 per port and for all ports	4
Outstanding requests for Tx descriptors per port	1	1	1	4 per port and for all ports	2
Outstanding requests for Rx descriptors per port	1	1	1	4 per port and for all ports	2
Credits for posted writes	4	4	4	4	4
Max payload size supported	512	512 B	512 B	512 B	512 B
Max request size supported	2 KB	2 KB	2 KB	2 KB	2 KB
Link layer retry buffer size	3.2 KB	3.2 KB	3.2 KB	3.2 KB	2 KB
Vital Product Data (VPD)	Y	Y[1]	N	Y	Y
VPD size	1024B	1024B[1]	N/A	256B	256B
End to End CRC (ECRC)	Y	Y	Y	Y	N
OBFF (Optimized Buffer Flush/Fill)	N	Y[2]	N[3]	N	N
Latency Tolerance Reporting (LTR)	Y	Y[2]	N[3]	Y	N

Table 1-6. Host Interface Features (Continued)

Feature	Foxville	Springville	Pearsonville	Powerville	82574
TPH	N	Y	Y	Y	N
CSR access via Configuration space	Y	Y	Y	Y	N
Access Control Services (ACS)	N	N	N	Y	N
Audio Video Bridging (AVB) support (in FDX mode only)	Y	Y	N[4]	N	N
TSN preemption, Time aware Shaper, Interspersing (in FDX mode only)	Y	N	N	N	N
PCIe PTM	Y	N	N	N	N

1. Not supported in Flash-less Foxville operation.
2. Feature fully functional in the device hardware. It is disabled by default via NVM due to the lack of OBFF enabled platforms at launch. Disabled by default via Flash.
3. Feature fully functional in the device hardware. It is disabled via NVM
4. Feature fully functional in the device hardware. It is not enabled by default driver settings.

Table 1-7. LAN Functions Features

Feature	Foxville	Springville	Pearsonville	Powerville	82574
Programmable host memory receive buffers	Y	Y	Y	Y	Y
Descriptor ring management hardware for transmit and receive	Y	Y	Y	Y	Y
ACPI register set and power down functionality supporting D0 and D3 states	Y	Y	Y	Y	Y
Software controlled reset bit (resets everything except the configuration registers)	Y	Y	Y	Y	Y
Software Definable Pins (SDPs) - per port	4	4	4	4	N
Four SDP pins can be configured as general purpose interrupts	Y	Y	Y	Y	N
Wake up	Y	Y	Y	Y	Y
Flexible wake-up filters	32	8	8	8	6
Flexible filters for queue assignment in normal operation	8	8	8	8	N
IPv6 wake-up filters	Y	Y	Y	Y	Y
Default configuration by the NVMFlash for all LEDs for pre-driver functionality	3 LED	3 LEDs	3 LEDs	4 LEDs	3 LEDs
LAN function disable capability	Y	Y	Y	Y	Y
Programmable memory transmit buffers	Y	Y	Y	Y	Y
Double VLAN	Y	Y	Y	Y	N
IEEE 1588	Y	Y	Y	Y	Y
Per-packet timestamp	Y	Y	Y	Y	N
Tx rate limiting per queue	Y	Y	N	N	N

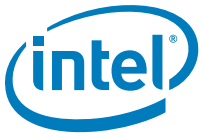


Table 1-8. LAN Performance Features

Feature	Foxville	Springville	Pearsonville	Powerville	82574
TCP segmentation offload - Up to 256 KB	Y	Y	Y	Y	Y
iSCSI TCP segmentation offload (CRC)	N	N	N	N	N
IPv6 support for IP/TCP and IP/UDP receive checksum offload	Y	Y	Y	Y	Y
Message Signaled Interrupts (MSI)	Y	Y	Y	Y	Y
Message Signaled Interrupts (MSI-X) number of vectors	5	5	5	25	5
Interrupt throttling control to limit maximum interrupt rate and improve CPU utilization	Y	Y	Y	Y	Y
Rx packet split header	Y	Y	Y	Y	Y
Receive Side Scaling (RSS)	Y	Y	Y	Y	Y
Total number of Rx queues per port	4	4	2	8	2
Total number of TX queues per port	4	4	2	8	2
RX header replication Low latency interrupt DCA support TCP timer interrupts No snoop Relax ordering	Yes to all but DCA	Yes to all	Yes to all	Yes to all	Only No snoop and Relax ordering
TSO interleaving for reduced latency	Y	Y	Y	Y	N
Receive Side Coalescing (RSC)	N	N	N	N	N
SCTP receive and transmit checksum offload	Y	Y	Y	Y	N
IPSec offload	N	N	N	N	N

Table 1-9. Manageability Features

Feature	Foxville	Springville	Pearsonville	Powerville	82574
Advanced pass-through-compatible management packet transmit/receive support	Y	Y[1]	N	Y	Y
Manageability support for ASF 1.0 and Alert on LAN 2.0	N	N	N	N	N
Manageability support for DMTF-DASH	N	N	N	N	N
Managed ports on SMBus interface to external MC	1[1]	1[1]	N	4	1
Auto-ARP reply over SMBus	Y	Y[1]	N	Y	Y
NC-SI Interface to an external MC	Y	Y[1]	N	Y	Y
Standard DMTF NC-SI protocol support	Y	Y[1]	N	Y	Y
DMTF MCTP protocol over SMBus	Y	Y[1]	N	Y	N
NC-SI hardware arbitration	Y	Y[1]	N	Y	N
DMTF MCTP protocol over PCIe	Y	Y[1]	N	N	N
KCS Host Interface	N	N	N	N	N
OS to BMC traffic	Y	Y[1]	N	Y	N
Internal IDE Redirection (IDER) Support	N	N	N	N	N
Internal Serial-Over-LAN (SoL) Support	N	N	N	N	N

Table 1-9. Manageability Features (Continued)

Feature	Foxville	Springville	Pearsonville	Powerville	82574
Intel® Advanced Management Technology System Defense Features: • Tx and Rx System Defense Feature Filters • System Defense Feature Headers Redirection	 Y N	 N N	 N N	 N N	 N N
Manageability L2 address filters	2	2	N	2	1
Manageability VLAN L2 filters	8	8	N	8	4
Manageability EtherType filters	4	4	N	4	N
Manageability Flex L4 port filters	8	8	N	8	4
Manageability Flex TCO filters	1	1	N	1	2
Manageability L3 address filters (IPv4)	4	4	N	4	1
Manageability L3 address filters (IPv6)	4	4	N	4	1
Proxying	1 ARP Offload 2 NS Offloads MLD support mDNS Offload	1 ARP Offload 2 NS Offloads MLD support mDNS Offload[1]	1 ARP Offload 2 NS Offloads MLD support	1 ARP Offload 2 NS Offloads	N

1. mDNS mDNS Offload is not supported in Flash-less Foxville operation.

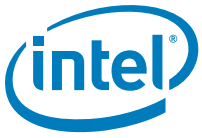
Table 1-10. Power Management Features

Feature	Foxville	Springville	Pearsonville	Powerville	82574
Magic packet wake-up enable with unique MAC address	Y	Y	Y	Y	Y
ACPI register set and power down functionality supporting D0 and D3 states	Y	Y	Y	Y	Y
Full wake-up support (APM and ACPI 2.0)	Y	Y	Y	Y	Y
Smart power down at S0 no link and Sx no link	Y	Y	Y	Y	Y
Ultra Low Power at Link Disconnected	Y	N	N	N	N
LAN disable functionality (equivalent to Static device off functionality)	Y	Y[1]	Y[1]	Y	Y
PCIe function disable	Y	Y	Y	Y	Y
Dynamic device off	Y	Y[2]	Y[2]	Y	Y
EEE	Y	Y	Y	Y	N
DMA coalescing	Y	Y	N[3]	Y	N
OBFF/PE_WAKE_N	N	Y[4]	N[5]	N	N
Integrated SVR / LVR control	Y	Y	Y		

1. Feature not functional if enabled together with dynamic device off.
2. Feature not functional if enabled together with static device off (such as LAN disable).
3. Feature fully functional in the device hardware. It is not enabled by default driver setting.
4. Feature fully functional in the device hardware. It is disabled by default via NVM due to the lack of OBFF enabled platforms at launch. Disabled by default in Flash due to the lack of OBFF enabled platforms at initial release.
5. Feature fully functional in the device hardware. It is not enabled by default driver setting.

1.6 Overview of Changes Compared to Springville

The following section describes the modifications designed in Foxville compared to Springville.



1.6.1 Link Speed - 2.5 Gb/s

Foxville supports the 2.5GBASE-T standard on top of its predecessors supporting link speeds up to 1 Gb/s.

1.6.2 PCIe Speed

Foxville Supports the PCIe Gen2 on top of Gen1supported by its predecessors

1.6.3 Manageability & Security

Foxville provides an interface to Intel management Engine - CSME. Together with the CSME, Foxville provides basic Circuit Breaker protection. For that purpose, the filters that monitor the LAN traffic on Foxville are enhanced compared to i210.

Foxville also introduces a mechanism to protect the external FLASH against wear-out.

1.6.4 Time Sensitive Networking

Foxville enhances its support in TSN applications supporting the following specifications:

- **Legacy**: 1588 - Basic time-sync support (supported by **Foxville** predecessor as well)
- **New**: 802.1AS-Rev - Higher precision time synchronization with multiple (dual) clock masters
- **Legacy**: 802.1Qav - Credit Based Shaping and Basic scheduling
- **New**: 802.1Qbv - Time Aware Shaper
- **New**: 802.1Qbu and 802.3br - Frame Preemption and Interspersing Express Traffic
- **New:** PCIe PTM for synchronization between the NIC and Host timers

1.7 Device Data Flows

1.7.1 Transmit Data Flow

Table 1-11 lists a high level description of all data/control transformation steps needed for sending Ethernet packets to the line.

Table 1-11. Transmit Data Flow

Step	Description
1	The host creates a descriptor ring and configures one of Foxville's transmit queues with the address location, length, head and tail pointers of the ring (one of 4 available Tx queues).
2	The host is requested by the TCP/IP stack to transmit a packet, it gets the packet data within one or more data buffers.
3	The host initializes descriptor(s) that point to the data buffer(s) and have additional control parameters that describe the needed hardware functionality. The host places that descriptor in the correct location at the appropriate Tx ring.
4	The host updates the appropriate queue tail pointer (TDT)
5	Foxville's DMA senses a change of a specific TDT and as a result sends a PCIe request to fetch the descriptor(s) from host memory.
6	The descriptor(s) content is received in a PCIe read completion and is written to the appropriate location in the descriptor queue internal cache.

Table 1-11. Transmit Data Flow (Continued)

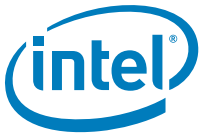
Step	Description
7	The DMA fetches the next descriptor from the internal cache and processes its content. As a result, the DMA sends PCIe requests to fetch the packet data from system memory.
8	The packet data is received from PCIe completions and passes through the transmit DMA that performs all programmed data manipulations (various CPU off loading tasks as checksum off load, TSO off load, etc.) on the packet data on the fly.
9	While the packet is passing through the DMA, it is stored into the transmit FIFO. After the entire packet is stored in the transmit FIFO, it is forwarded to the transmit switch module.
10	The transmit switch arbitrates between host and management packets and eventually forwards the packet to the MAC.
11	The MAC appends the L2 CRC to the packet and sends the packet to the line using a pre-configured interface.
12	When all the PCIe completions for a given packet are done, the DMA updates the appropriate descriptor(s).
13	After enough descriptors are gathered for write back or the interrupt moderation timer expires, the descriptors are written back to host memory using PCIe posted writes. Alternatively, the head pointer can only be written back.
14	After the interrupt moderation timer expires, an interrupt is generated to notify the host device driver that the specific packet has been read to Foxville and the driver can release the buffers.

1.7.2 Receive Data Flow

Table 1-12 lists a high level description of all data/control transformation steps needed for receiving Ethernet packets.

Table 1-12. Receive Data Flow

Step	Description
1	The host creates a descriptor ring and configures one of Foxville's receive queues with the address location, length, head, and tail pointers of the ring (one of 4 available Rx queues).
2	The host initializes descriptors that point to empty data buffers. The host places these descriptors in the correct location at the appropriate Rx ring.
3	The host updates the appropriate queue tail pointer (RDT).
4	Foxville's DMA senses a change of a specific RDT and as a result sends a PCIe request to fetch the descriptors from host memory.
5	The descriptors content is received in a PCIe read completion and is written to the appropriate location in the descriptor queue internal cache.
6	A packet enters the Rx MAC. The Rx MAC checks the CRC of the packet.
7	The MAC forwards the packet to an Rx filter.
8	If the packet matches the pre-programmed criteria of the Rx filtering, it is forwarded to the Rx FIFO. VLAN and CRC are optionally stripped from the packet and L3/L4 checksum are checked and the destination queue is fixed.
9	The receive DMA fetches the next descriptor from the internal cache of the appropriate queue to be used for the next received packet.
10	After the entire packet is placed into the Rx FIFO, the receive DMA posts the packet data to the location indicated by the descriptor through the PCIe interface. If the packet size is greater than the buffer size, more descriptors are fetched and their buffers are used for the received packet.
11	When the packet is placed into host memory, the receive DMA updates all the descriptor(s) that were used by packet data.
12	After enough descriptors are gathered for write back or the interrupt moderation timer expires or the packet requires immediate forwarding, the receive DMA writes back the descriptor content along with status bits that indicate the packet information including what off loads were done on that packet.
13	After the interrupt moderation timer completes or an immediate packet is received, Foxville initiates an interrupt to the host to indicate that a new received packet is already in host memory.
14	Host reads the packet data and sends it to the TCP/IP stack for further processing. The host releases the associated buffers and descriptors once they are no longer in use.

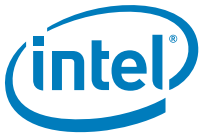


2.0 Pin Interface

Foxville is a 56-pin, 7x7mm QFN package. A top level view of the pin layout is illustrated below in . The IO pins are also listed in Table 2-1.

Table 2-1. Foxville Pin List

Pin Num	Pin Name	Internal weak Pull Up/Down on Reset	Internal weak Pull Up/Down during nominal operation		Pin Num	Pin Name	Internal weak Pull Up/Down on Reset	Internal weak Pull Up/Down during nominal operation
1	LAN_PWR_GOOD	PU	PU		29	JTAG_TCK	PU	PU
2	LAN_DISABLE_N	PU	PU		30	LED2	PU	
3	PE_CLKREQ_N	PU			31	LED1	PU	
4	ULP_WAKE_N	PU			32	LED0	PU	
5	+0.9V_CTRL				33	+3.3V_PAD		
6	+0.9V_CTRL				34	SPI_MOSI	PU	PU
7	+3.3V_DC				35	SPI_CLK	PU	PU
8	VQPS				36	SPI_DIN	PU	PU
9	+0.9V_CORE				37	SPI_CS_N	PU	PU
10	SMB_DATA	PU			38	+0.9V_CORE		
11	SMB_CLK	PU			39	+3.3V_XO		
12	PE_WAKE_N	PU			40	XTAL_OUT		
13	PE_RST_N	PU	PU		41	XTAL_IN		
14	+3.3V_PAD				42	PHY_CAL		
15	SDP0	PU	PU		43	+3.3V_CDB		
16	JTAG_RST	PD	PD		44	+0.9V_A		
17	JTAG_TMS / SDP2	PU	PU		45	MDI_A_P		
18	JTAG_TDO / SDP3	PU	PU		46	MDI_A_N		
19	JTAG_TDI / SDP1	PU	PU		47	+3.3V_A		
20	RBIAS				48	MDI_B_P		
21	PE_CLK_P				49	MDI_B_N		
22	PE_CLK_N				50	+0.9V_A		
23	+3.3V_VPH				51	MDI_C_P		
24	PE_TX_N				52	MDI_C_N		
25	PE_TX_P				53	+3.3V_A		
26	+0.9V_VP				54	MDI_D_P		
27	PE_RX_N				55	MDI_D_N		
28	PE_RX_P				56	+0.9V_A		



2.1 Signal Type Definition - Abbreviations

Table 2-1 below summarizes the abbreviations used in this chapter for the IO pins and buffers' types.

Table 2-2. Pin and Buffer Type's Abbreviations

Pin Type Abbreviations	Description		Buffer Type Abbreviations	Description
I	Input-only, digital levels			
O	Output-only, digital levels			
I/O	Bidirectional input/output signal, digital levels			
Prg	Bidirectional pad, programmable to operate either as input or output, digital levels		PWR	Power supply pins
AI	Input-only, analog levels		GND	Ground
AO	Output-only, analog levels		A	Analog characteristics, see the AC/DC specification for more detail.
AI/O	Bidirectional, analog levels		Dig	Digital Input or output signal
PWR	Power		Prg	Digital programmable pin with an alternate function
GND	Ground		OD	Open drain signal

2.2 Ethernet Media Interface

Table 2-3. Ethernet Media Interface

Pin No.	Pin Name	Pin Type	Buffer Type	Functionality for MDI_LANE_SWAP = '1'	Functionality for MDI_LANE_SWAP = '0'	Connection
45	**MDI_A_P**	AI / AO	A	MDI_A_P	MDI_D_N	**Transmit/Receive Positive/Negative** Connect directly to XFMR without any pull-down terminators, such as resistors or capacitors, required
46	**MDI_A_N**	AI / AO	A	MDI_A_N	MDI_D_P	
48	**MDI_B_P**	AI / AO	A	MDI_B_P	MDI_C_N	
49	**MDI_B_N**	AI / AO	A	MDI_B_N	MDI_C_P	
51	**MDI_C_P**	AI / AO	A	MDI_C_P	MDI_B_N	
52	**MDI_C_N**	AI / AO	A	MDI_C_N	MDI_B_P	
54	**MDI_D_P**	AI / AO	A	MDI_D_P	MDI_A_N	
55	**MDI_D_N**	AI / AO	A	MDI_D_N	MDI_A_P	
42	**PHY_CAL**	AI / AO	A	**Calibration for all GPHY Ethernet Port**		

2.3 PCIe Interface

Table 2-4. PCIe Interface

Pin No.	Pin Name	Pin Type	Buffer Type	Function
28	**PE_Rx_P**	AI	A	**Differential PCIe Data Input Pair** These are the negative and positive signals respectively of the differential input pair of the PCIe interface. The pair samples up to 8 Gbit/s differential data signal for PCIe. These pins must be AC-coupled.
27	**PE_Rx_N**	AI	A	

Table 2-4. PCIe Interface

Pin No.	Pin Name	Pin Type	Buffer Type	Function
25	**PE_Tx_P**	AO	A	**Differential PCIe Data Output Pair** These are the negative and positive signals respectively of the differential output pair of the PCIe interface. The pair samples up to 8 Gbit/s differential data signal for PCIe. These pins must be AC-coupled.
24	**PE_Tx_N**	AO	A	
20	**RBIAS**	AI/O	A	**Pad to connect external tuning resistor**
21	**PE_CLK_P**	AI	A	**Differential PCIe Clock Pair** These are the negative and positive signals respectively of the differential input clock pair of the PCIe interface. The input clock is 100 MHz external reference clock used by PHY.
22	**PE_CLK_N**	AI	A	
13	**PE_RST_N**	I	Prg	**PCIe Reset** Power and Clock Good Indication. The PE_RST_N signal indicates that both PCIe power and clock are available.
12	**PE_WAKE_N**	I/O	OD	**PCIe Wake** P31G drives this signal to zero when it detects a wakeup event and either:• the PME_en bit in PMCSR is 1b or• the APME bit of the Wake Up Control (WUC) register is 1b.
3	**PE_CLKREQ_N**	I/O	OD	**PCIe Clock Request** PCIe CLKREQ# for power management. This pin is used to wakeup from L1 sub-states to L0.

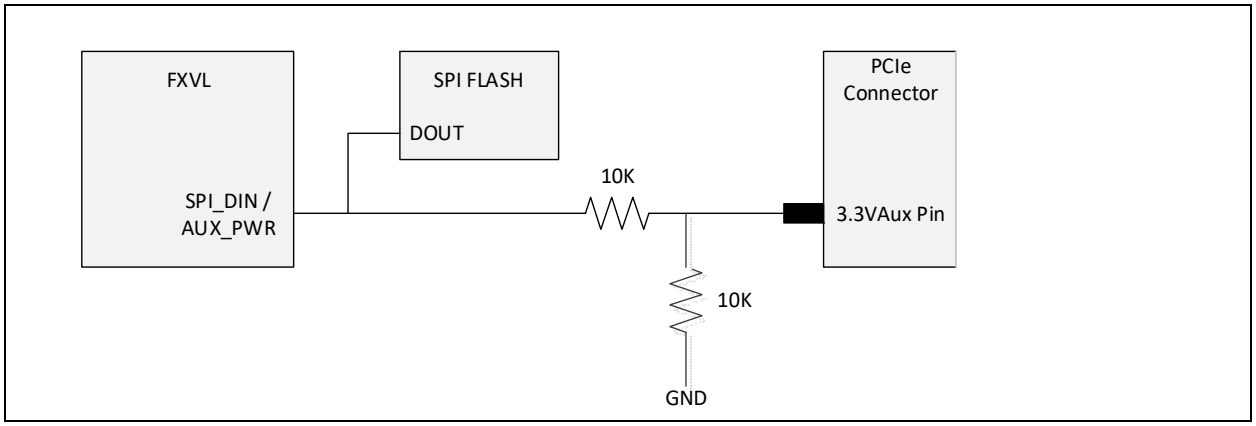


Figure 2-1. AUX_PWR straping connection

2.4 Management and SPI Flash Interfaces

2 types of serial management interfaces are provided: SPI master interface and MDIO slave interface.

Table 2-5. Management Interfaces

Pin No.	Pin Name	Pin Type	Buffer Type	Function
MDIO Slave Interface				
4	**ULP_WAKE_N**	I/O	OD	**ULP Wake** This pin is used by the external Management controller (CSME) and the host system to wake the device from ULP. For that matter, this signal should be connected as described in the following equation: ULP_WAKE_N = SMB_DATA && not (PE_RST_N)
11	**SMB_CLK (SML0B_CLK)**	I/O	OD	**SMBus Clock** One clock pulse is generated for each data bit transferred. An external pull-up resistor of 499/2.2K/10K ohm is required when setting the SMBus for 1M/400K/100K Hz respectively.
10	**SMB_DATA (SML0B_DATA)**	I/O	OD	**SMBus Data** Stable during the high period of the clock (unless it is a start or stop condition). An external pull-up resistor is required as the SMB_CLK.
SPI Master interface				
34		O	Prg	**SPI Data Output (Foxville Out Flash In)** SPI interface data output
36	**SPI_DIN / AUX_PWR**	I	Prg	**SPI Data Input (Foxville In Flash Out)** This pin is also the AUX_PWR straping. External pull-up / pull-down are not required in systems on which the device gets power during D3cold. In NIC designs this pin should be connected as shown in Figure 2-2.
35	**SPI_CLK**	O	Prg	**SPI Clock** During nominal operation this pin should be connected to an external 10K ohm pull down resistor.
37	**SPI_CS_N / PCIE_ENA**	O	Prg	**SPI Chip Select** Active low signal. This pin is also the PCI Function Enable straping pin when sampled at high on reset.

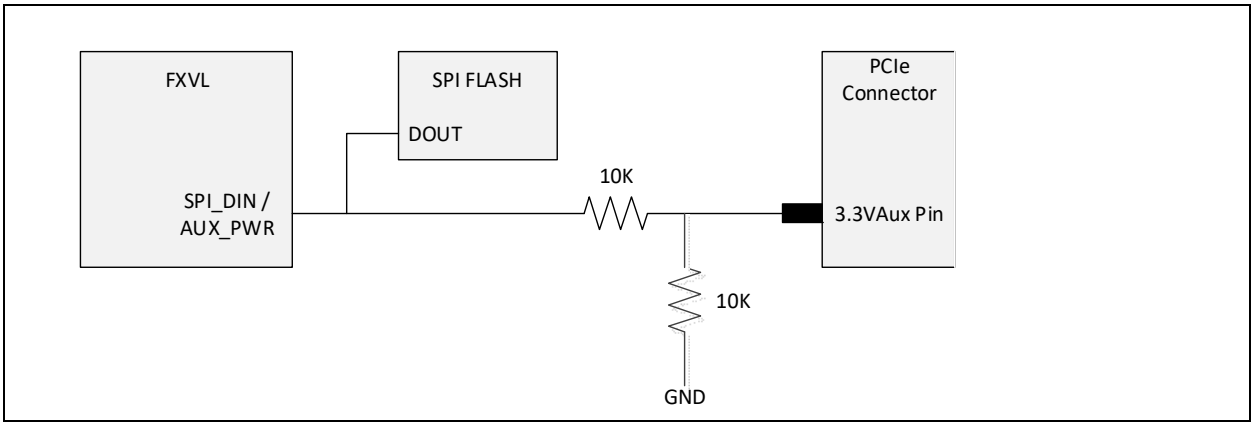


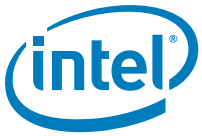
Figure 2-2. AUX_PWR straping connection

2.5 LED / JTAG / UART Interface

The LED interface is used to connect external LEDs for Ethernet status indication of the Ethernet PHY interfaces. Single and dual color LEDs are supported.

Table 2-6. LED/UART/JTAG Interface

Pin No.	Pin Name	Pin Type	Buffer Type	Function
32	**LED0**	O	Prg	**LED0** LED control output, freely configurable, drives single-color or dual color LEDs.
31	**LED1**	O	Prg	**LED1** LED control output, freely configurable, drives single-color or dual color LEDs.
30	**LED2**	O	Prg	**LED2** LED control output, freely configurable, drives single-color or dual color LEDs.
29	**JTAG_TCK**	I	Dig	**JTAG Test Clock** The signals TDI, TDO and TMS are synchronous subject to this JTAG test clock. If JTAG Controller is held in reset state, the device operates in normal mode, this clock pin does not need to be clocked.
16	**JTAG_RST**	I	Dig	**JTAG Test Reset** At Low the JTAG is in the reset state and the multiplexed pins JTAG/SDPs act as SDPs. This pin has an internal weak pull-down that holds the JTAG controller in its reset state if the pin is left open. This is a difference to the JTAG specification given by IEEE 1149.1.
17	**JTAG_TMS / SDP2**	I / Prg	Prg	**JTAG Test Mode Select /** **Software Defined Pin 2** Software defined pin 2. It can be selected as input or output mode.
18	**JTAG_TDO / SDP3 / MDI_LANE_SWAP**	O / Prg	Prg	**JTAG Serial Test Data Output /** **Software Defined Pin 3** Software defined pin 3. It can be selected as input or output mode. This pin is the straping pin. It has an internal weak pull up and should be connected to an external 10K ohm pull down resistor if needed to be sampled at low level. At high, select the RJ45 TAP positive option and at low it selects the RJ45 TAP negative option.
19	**JTAG_TDI / SDP1**	I / Prg	Prg	**JTAG Serial Test Data Input /** **Software Defined Pin 1** Software defined pin 1. It can be selected as input or output mode.



2.6 Miscellaneous Signals

Table 2-7. Miscellaneous Signals

Pin No.	Pin Name	Pin Type	Buffer Type	Function
41	**XTAL_IN**	AI	A	**Crystal: Oscillator Input** A crystal must be connected between XTAL1 and XTAL2. Additional Load Capacitances must tie both pins also the GND.
	CLK	I		**Clock: Clock Input** The clock must have a frequency accuracy of +/-50ppm.
40	**XTAL_OUT**	AO	A	**Crystal: Oscillator Output** A crystal must be connected between XTAL1 and XTAL2. Additional Load Capacitances must tie both pins also the GND.
15	**SDP0 (SW_DP)**	I/O	Prg	**Software Defined Pin 0** Software defined pin 0. It can be selected as input or output mode.
	CLKO	O		**Clock: Clock Output** Forwarding crystal oscillator clock to this output. The clock must have a frequency accuracy of +/-50ppm.
1	**LAN_PWR_GOOD**	I	Dig	**LAN Power Good** A 3.3V input signal. A transition from low to high initializes the device into operation. If the internal Power-on-Reset (POR) circuit is used to trigger device power-up, this signal should be connected to VDDP.
2	**LAN_DISABLE_N**	I	Dig	**Device Off** This is a 3.3V input signal. Asserting LAN_DISABLE_N puts Foxville in device disable mode. Note that this pin is asynchronous. Functionality of this input can be changed by NVM bits settings
5, 6	**+0.9V_CTRL**	AO	A	**DCDC Regulator Output** This is the group of pins providing a current output to self-supply a nominal voltage of *0.9V* with a worst case tolerance ±10%.

2.7 Power Supply

Table 2-8. Power Supply

Pin No.	Pin Name	Pin Type	Function
47, 53	**+3.3V_A**	PWR	**High-Voltage Domain Supply** This is the group of supply pins for the high voltage domain. It supplies the AFE of the Gigabit Ethernet PHY. This supply has to provide a nominal voltage of V_{DDA3V3}=*3.3V* with a worst case tolerance ±10% at the corners, respectively. *Note: For optimal power-consumption figure of merits the lowest possible voltage shall be selected in the system.*
44, 50, 56	**+0.9V_A**	PWR	**Low-Voltage Domain Supply** This is the group of supply pins for the low voltage domain. It supplies mixed signal blocks in the AFE and CDB of the Gigabit Ethernet PHY. This supply has to provide a nominal voltage of V_{DDA0V9}=*0.9V* with a worst case tolerance ±10%. *Note:*
39	**+3.3V_XO**	PWR	**XO Pad-Voltage Domain P Supply** This is the group of supply pins for the pad-supply of ROPLL and XO. This supply has to provide a nominal voltage of $V_{DDA3V3XO}$=*3.3V* with a worst case tolerance ±10% at the corners, respectively. *Note: For optimal power-consumption figure of merits the lowest possible voltage shall be selected in the system.*
43	**+3.3V_CDB**	PWR	**CDB High-Voltage Domain Supply** This is the group of supply pins for the CDB. This supply has to provide a nominal voltage of $V_{DDA3V3CDB}$=*3.3V* with a worst case tolerance ±10% at the corners, respectively. *Note: For optimal power-consumption figure of merits the lowest possible voltage shall be selected in the system.*
26	**+0.9V_VP**	PWR	**PCIe Low-Voltage Domain Supply** This is the group of supply pins for the low voltage domain of the PCIe interface. It supplies mixed signal blocks in the PMA of the PCIe interface. This supply has to provide a nominal voltage of V_P=*0.9V* with a worst case tolerance ±10%. *Note:*
23	**+3.3V_VPH**	PWR	**PCIe High-Voltage Domain Supply** This is the group of supply pins for the high voltage domain of the PCIe interface. It supplies mixed signal blocks in the PHY of the PCIe interface. This supply has to provide a nominal voltage of V_{PH}=*3.3V* with a worst case tolerance ±10%. *Note:*
14, 33	**+3.3V_PAD**	PWR	**Pad-Voltage Domain Supply** This is the group of supply pins for the pad-supply of the Gigabit Ethernet PHY. This supply has to provide a nominal voltage of V_{DDP}=*3.3V* with a worst case tolerance ±10% at the corners, respectively. *Note: For optimal power-consumption figure of merits the lowest possible voltage shall be selected in the system.*
9, 38	**+0.9V_CORE**	PWR	**Core-Voltage Domain Supply** This is the group of supply pins for the core digital voltage domain. It supplies the digital core blocks of the Gigabit Ethernet PHY. This supply has to provide a nominal voltage of V_{DD}=*0.9V* with a worst case tolerance ±10%. *Note:*

Table 2-8. Power Supply

Pin No.	Pin Name	Pin Type	Function
8	**Reserved (VQPS)**	PWR	During nominal operation this pin should be tight to ground.
7	**+3.3V_DC**	PWR	**DCDC Power Supply** This is the group of supply pins for the DCDC. This supply has to provide a nominal voltage of $V_{DD3V3DCDC}$=*3.3V* with a worst case tolerance ±10%. *Note:*
EPAD	VSS	GND	General Device Ground The EPAD is the exposed pad at the bottom of the package. This pad must be properly connected to the ground plane of the PCB.

2.7.1 Leakage Avoidence on LAN power disconnect

Normally users will use LAN_DISABLE_n pin to disable Foxville. However there are some cases where users prefer to use a FET on the 3.3V LAN power supply and totally power-down the device.

When customer disable LAN_VCC (+3.3V_LAN), there is a leakage to happen on 5 Foxville Pins:

- Foxvile Pin 3: PE_CLKREQ_N
- Foxville Pin 4: ULP_WAKE_N
- Foxville Pin 12: PE_WAKE_N
- Foxville Pin 10: SMB_DATA
- Foxville Pin 11: SMB_CLK

If OEMs insist to connect Foxville VCC to primary power rail but DSW and only agree to disable LAN by shutting down LAN_VCC, **OEMs can design extra isolation circuit (reference as below)**.

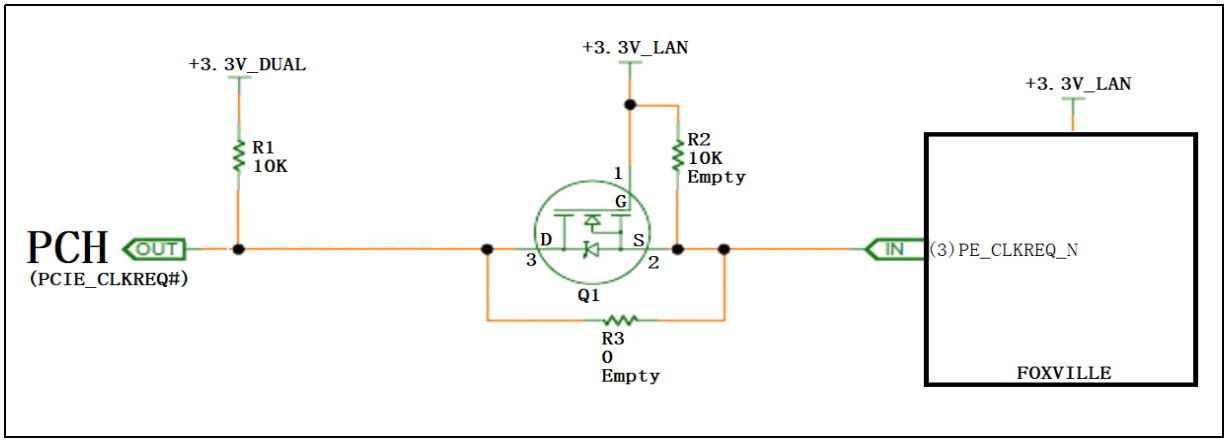


Figure 2-3. Leakage protection circuit

Component: The main active component will be the general purpose N-ch FET (Q1)

Why R2 empty? Foxville pin#3 already has an internal pull-up. Therefore, R2 is only a reserve.

Why R3 empty? This is a reserved bypass path. It is only for testing/validation when needed.

Logic:

When +3.3V_LAN is shut down → Q1 Vgs=0 → Q1 is off. No leakage into Foxville Pin#3.

When +3.3V_LAN is powered on:

If Foxville Pin#3 is low → Q1 Vgs=3.3V → Q1 is on → Foxville pin#3 will drive the PCH's input = low.

If Foxville Pin#3 is high → Q1 Vgs=0V → Q1 is off → R1 will pull up PCH's input = high.

3.0 Interconnects

3.1 PCIe

3.1.1 PCIe Overview

PCIe is a third generation I/O architecture that enables cost competitive I/O solutions providing industry leading price/performance and features. It is an industry-driven specification.

PCIe defines a basic set of requirements that encases the majority of the targeted application classes. Higher-end applications' requirements, such as enterprise class servers and high-end communication platforms, are encased by a set of advanced extensions that compliment the baseline requirements.

To guarantee headroom for future applications of PCIe, a software-managed mechanism for introducing new, enhanced, capabilities in the platform is provided. Figure 3-1 shows PCIe architecture.

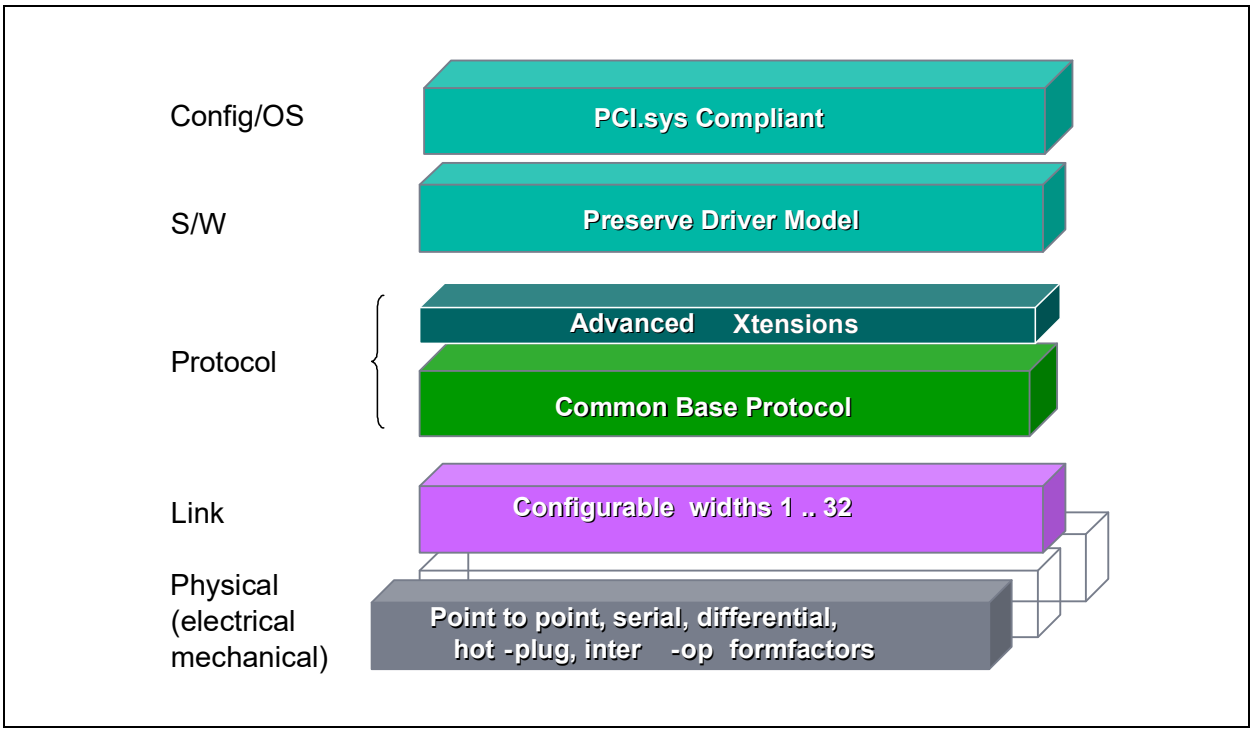


Figure 3-1. PCIe Stack Structure

PCIe's physical layer consists of a differential transmit pair and a differential receive pair. Full-duplex data on these two point-to-point connections is self-c such that no dedicated clock signals are required. The bandwidth of this interface increases linearly with frequency.

The packet is the fundamental unit of information exchange and the protocol includes a message space to replace the various side-band signals found on many buses today. This movement of hard-wired signals from the physical layer to messages within the transaction layer enables easy and linear physical layer width expansion for increased bandwidth.

The common base protocol uses split transactions and several mechanisms are included to eliminate wait states and to optimize the reordering of transactions to further improve system performance.

3.1.2 General Functionality

3.1.2.1 Native/Legacy

All Foxville PCI functions are native PCIe functions.

3.1.2.2 Transactions

Foxville does not support requests as target or master.

3.1.3 Host Interface

3.1.3.1 Tag IDs

PCIe device numbers identify logical devices within the physical device (Foxville is a physical device). Foxville implements a single logical device with PCI function. The device number is captured from type 0 configuration write transaction.

The PCIe function interfaces with the PCIe unit through one or more clients. A client ID identifies the client and is included in the *Tag* field of the PCIe packet header. Completions always carry the tag value included in the request to enable routing of the completion to the appropriate client.

Tag IDs are allocated differently for read and write. Messages are sent with a tag of 0x0.

3.1.3.1.1 TAG ID Allocation for Read Transactions

Table 3-1 lists the Tag ID allocation for read accesses. The tag ID is interpreted by hardware in order to forward the read data to the required device.

Table 3-1. IDs in Read Transactions

Tag ID	Description	Comment
0x0	Data request 0	
0x1	Data request 1	
0x2	Data request 2	
0x3	Data request 3	
0x4	Data request 4	
0x5	Data request 5	

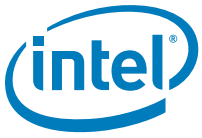


Table 3-1. IDs in Read Transactions

0x6-017	Not used	
0x18	Descriptor Tx	
0x19 -0x1B	Not used	
0x1C	Descriptor Rx	
0x1D -0x1F	Not used	

3.1.3.1.2 Priority TAG for Read and Write Transactions

The PCIe header contains a priority Tag for the transaction over the PCIe bus. This priority Tag provides a hint to the system for the priority the PCIe packet should be handled. In some systems, PCIe packets with higher priority Tag would get precedence over PCIe packets with lower priority Tag.

Foxville provides a programmable priority Tag for PCIe transactions related to transmit and receive queues. The priority level is configured by the PCIe_PRIO field in the RXCTL and TXCTL registers for the receive and transmit queues receptively. other PCIe transaction carry a priority Tag equals to zero.

3.1.3.2 Completion Timeout Mechanism

In any split transaction protocol, there is a risk associated with the failure of a requester to receive an expected completion. To enable requesters to attempt recovery from this situation in a standard manner, the completion timeout mechanism is defined.

The completion timeout mechanism is activated for each request that requires one or more completions when the request is transmitted. Foxville provides a programmable range for the completion timeout, as well as the ability to disable the completion timeout altogether. The completion timeout is programmed through an extension of the PCIe capability structure (refer to Section 9.4.5.12).

Foxville's reaction in case of a completion timeout is listed in Table 3-18.

Foxville controls the following aspects of completion timeout:

- Disabling or enabling completion timeout.
- Disabling or enabling re-send of a request on completion timeout.
- A programmable range of re-sends on completion timeout, if re-send enabled.
- A programmable range of timeout values.
- Programming the behavior of completion timeout is listed in Table 3-2.

Table 3-2. Completion Timeout Programming

Capability	Programming capability
Completion Timeout Enabling	Controlled through *PCI Device Control 2* configuration register.
Resend Request Enable	Loaded from the NVMinto the *GCR* register.
Number of Re-sends on Timeout	Controlled through *GCR* register.
Completion Timeout Period	Controlled through *PCI Device Control 2* configuration register.

Completion Timeout Enable - Programmed through the PCI Device Control 2 configuration register. The default is: Completion Timeout Enabled.

Resend Request Enable - The *Completion Timeout Resend* NVM bit (loaded to the *Completion_Timeout_Resend* bit in the PCIe Control (GCR) register enables resending the request (applies only when completion timeout is enabled). The default is to resend a request that timed out.

Number of re-sends on timeout - Programmed through the *Number of resends* field in the GCR register. The default value of resends is 3.

3.1.3.2.1 Completion Timeout Period

Programmed through the PCI Device Control 2 configuration register (refer to Section 9.4.5.12). Foxville supports all ranges defined by PCIe Gen2 v3.1.

A memory read request for which there are multiple completions are considered completed only when all completions have been received by the requester. If some, but not all, requested data is returned before the completion timeout timer expires, the requestor is permitted to keep or to discard the data that was returned prior to timer expiration.

Note: The completion timeout value must be programmed correctly in PCIe configuration space (in the Device Control 2 register); the value must be set above the expected maximum latency for completions in the system in which Foxville is installed. This ensures that Foxville receives the completions for the requests it sends out, avoiding a completion timeout scenario. It is expected that the system BIOS sets this value appropriately for the system.

3.1.4 PCI PTM Support

Foxville supports the PCI PTM protocol by which software can get an indication of the time difference between the host timer and Foxville timers. Time difference between the timers is measured by a similar concept to PTP. The PTM sequence is illustrated in Figure 3-2.

Foxville initiates the PTM sequence by sending a PTM request message on the PCIe to the downstream port of the host. In return it gets back the PTM Response or PTM ResponseD messages. The following subsections describe the PTM state machine and the methods the PTM sequences are initiated.

Both Foxville and the host sample the time that the PTM messages are sent. The host provides the time of the messages transmitted by the host with the PTM message as shown in Figure 3-2.

Assuming that the pace of the host timer and Foxville timer are almost the same, and assuming that the transmission latencies on both direction are the same, then...

$$T_{delay} = \text{PCIe Transmission Latency} = 0.5 * [(T4 - T1) - (T3 - T2)]$$
$$\Delta T = T_{nic} - T_{host} = 0.5 * [(T1 + T4) - (T2 + T3)]$$

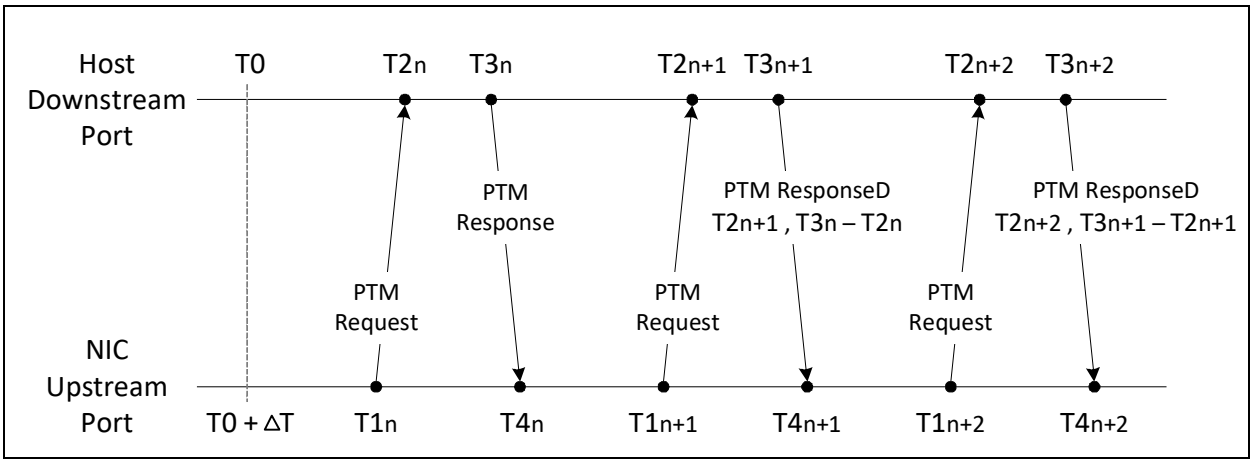


Figure 3-2. PTM Sequence (Ideal Case)

3.1.4.1 PTM Messages Latency and Accuracy

In practice, Foxville samples the time of the transmitted PTM messages before they are actually transmitted on the PCIe bus. Similarly, Foxville samples the time of the received PTM messages after they are received on the PCIe bus. As opposed to the transmit and receive latencies on the PCIe bus that are assumed to be symmetric, the internal transmit and receive latencies between the sampling point and the PCie bus are not symmetric (as shown in Table 3-3). Note that these latencies values should be programmed by software. A timing diagram that takes into account the internal transmit and receive latencies is shown in Figure 3-3. Then the actual corrected equations that software should calculate for the T delay and delta T (ΔT) are described in Table 3-4.

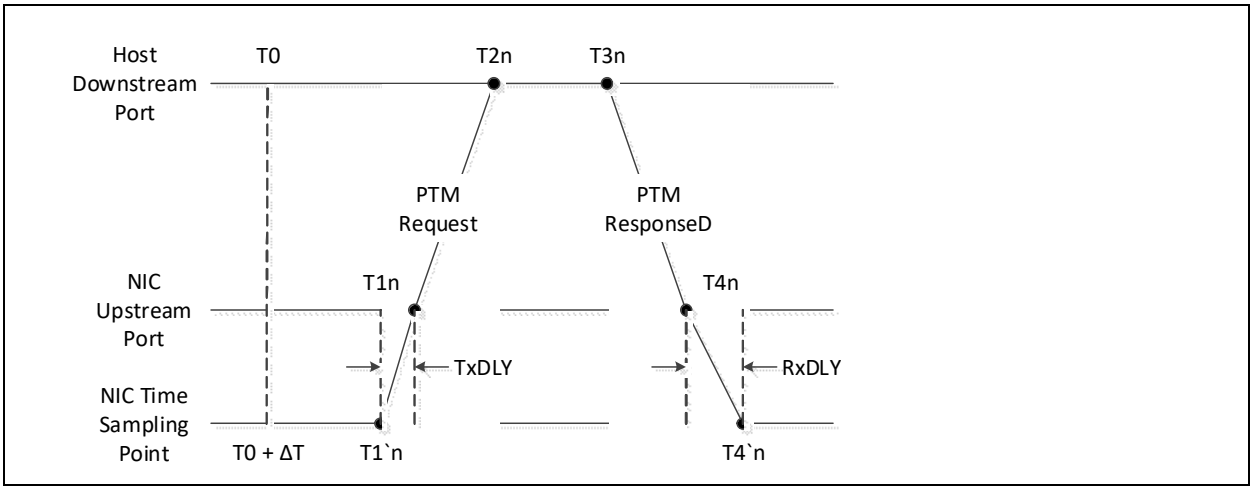


Figure 3-3. PTM Sequence (Actual Case)

Table 3-3. PTM Time-Stamping Latencies on the PCIe

Parameter (*PHY* + *DIG*)	Description	Delay Precision Range	PCIe_*PHY*_Delay setting by Software	PCIe_*DIG*_Delay setting by Software
TxDLY	Tx Delay on PCIe Gen1 / Gen2	+-10 nsec/ +- 6 nsec	0x0 / 0x0	0x0 / 0x0
RxDLY	Rx Delay on PCIe Gen1 / Gen2	+- 18 nsec / +- 10 nsec	0x90 / 0x40	0x44 / 0x01

Using the notations in Figure 3-3 then: T1 = T1‘ + TxDLY and T4 = T4‘ - RxDLY

Taking into account the TxDLY and RxDLY we get the actual T delay and ΔT parameters:

Table 3-4. Actual T delay and ΔT parameters including internal time-sampling latencies

T delay = PCIe Transmission Latency = 0.5 * [(T4‘-T1‘) - (T3-T2) - TxDLY - RxDLY]
ΔT = T nic - T host = T1‘ - T2 + 0.5 * [(T4‘ - T1‘) - (T3 - T2) + TxDLY - RxDLY)]

3.1.4.2 PTM Messages

The PTM Messages on the PCIe bus are shown in Table 3-5, Table 3-6 and Table 3-7. These format of these messages is copied from the PCIe PTM standard revision 1.0a. For more details please see the standard.

Table 3-5. Precision Time Measurement Messages

Message Name	TLP Type	Message Code	Routing[2:0]	Support RC EP SW BR	Description
PTM Request	Msg	0101 0010 b	100 b	r t tr	Initiate PTM Sequence
PTM Response	Msg	0101 0011 b	100 b	t r tr	Completes the PTM Sequence with no timing information
PTM ResponseD	MsgD	0101 0011 b	100 b	t r tr	Completes the PTM Sequence with timing information

Table 3-6. PTM Request / Response Messages

	+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
Byte 0	Fmt 001			Type					R	TC			R	Att	R	TH	TD	EP	Att		R				Reserved							
Byte 4	Requester ID																Tag								Message Code							
Byte 8	Reserved																															
Byte 12	Reserved																															

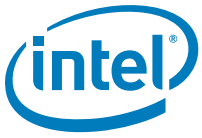


Table 3-7. PTM ResponseD Message

	+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
Byte 0	Fmt 001			Type					R	TC			R	Att	R	TH	TD	EP	Att		R				Length = 0x1							
Byte 4	Requester ID																Tag								Message Code							
Byte 8	MSB . . . . . PTM Master Time [63:32] . . . . LSB																															
Byte 12	MSB . . . . . PTM Master Time [31:0] . . . . LSB																															
Byte 16	MSB . . . . . Propagation Delay [31:0] = T3 - T2 . . . . LSB																															

In a later revision of the PCIe PTM spec, the "Propagation Delay" field is defined explicit in Big-Endian notation (as shown in the above Table 3-7). Unfortunately, there are already components in the market that relates to the "Propagation Delay" field in Little-Endian notation. Foxville tolerates both little and big endian notations of the "Propagation Delay". It is controlled by the "PTM Endian Notation" field in the NVM as follow:

"PTM Endian Notation" setting	Reported T3-T2 parameter in the "PTM Message"
Little Endian	The reported T3-T2 parameter in the "PTM Message" equals to the PTM_Prev_T3m2 register
Big Endian	The reported T3-T2 parameter in the "PTM Message" is a reversed byte ordering of the PTM_Prev_T3m2 register
Unknown	Foxville auto-detect the endianess notation: If the higher 16 bits of the PTM_Prev_T3m2 register are zero's, it is assumed to be little endian and it is reported as is in the "PTM Message". Else if the lower 16 bits of the PTM_Prev_T3m2 register are zero's, it is assumed to be big endian and its reversed bytes ordering is reported in the "PTM Message" Else if both the lower and higher 16 bits of the PTM_Prev_T3m2 register are non-zero then the PTM_Prev_T3m2 register is reported as is in the "PTM Message" with an "ambiguous Endianess" error indication in the message status.

3.1.4.3 PTM State Machine

The Figure 3-4 below describes Foxville's PTM state machine. The simplified explanation (for non erroneous cases) is described by the following steps:

- At init time the state machine wakes up with not timing information
- Foxville initiates the PTM Request message and samples its transmission time = T1.
- Following the completion of the first PTM cycle (getting back the PTM Response message) the state machine holds onlt the T1 parameter.
- Following the completion of the first PTM cycle (getting back the PTM Response message) the state machine holds also the T2 parameter.
- Following the completion of any further PTM cycles (getting back the PTM Response message) the state machine holds all needed timing parameters: T1, T2, T3-T2 and T4-T1. Foxville compute the Tdelay and the ΔT and post it in a packet to host memory.

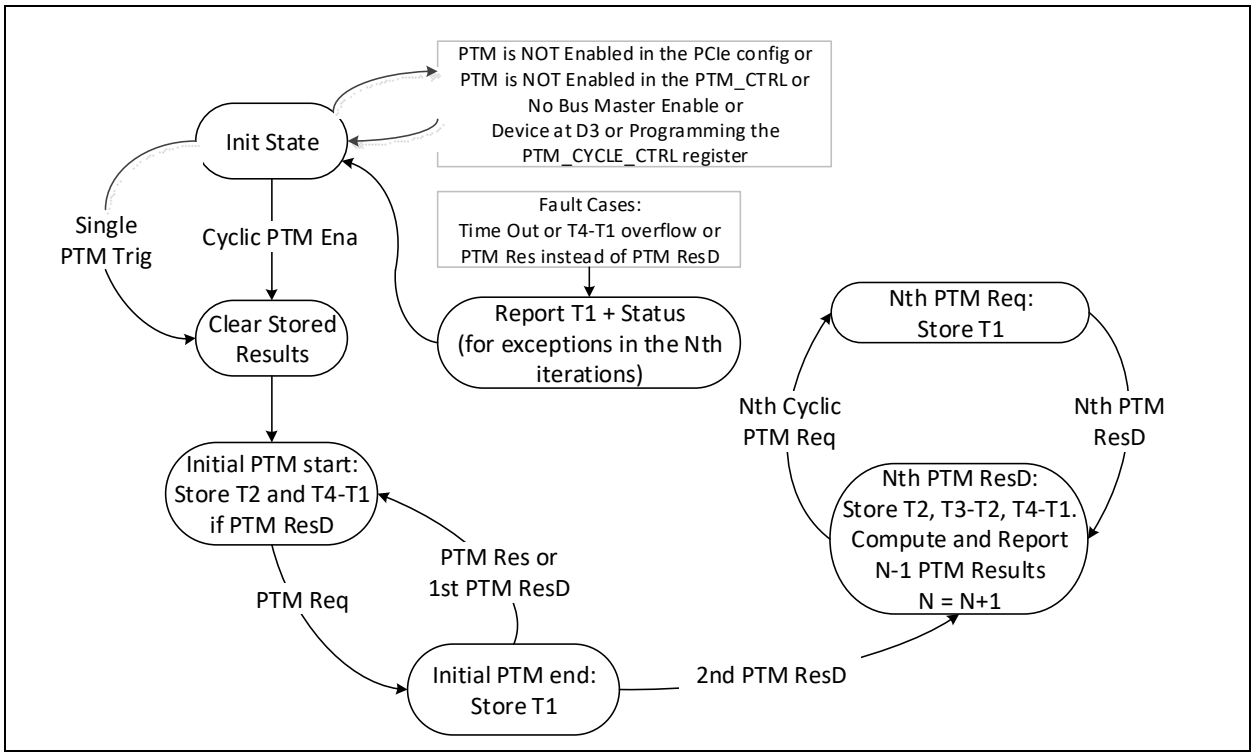


Figure 3-4. PTM State Machine

3.1.4.4 PTM Cycles Initiation

The PTM cycles can be initiated by one of two mutually exclusive methods:

- Explicit software initiation by setting the PTM_Trig flag in the PTM_CTRL register. Note that software should clear this flag triggering the next PTM cycles.
- Periodic initiation of the PTM cycles is enabled by the PTM_CYCLE_CTRL register.

3.1.4.5 Exceptions Handling

The Table 3-8 below summarizes possible exception handling.

Table 3-8. PTM Exceptions

Test Condition	Exception	Current implementation	Comment
Single Step	Time Out or PTM Res instead of PTM ResD	Return to Init state. No log in STATUS and No report msg to MNG. Need to set the Single Step to start again. No need to clear the STATUS.	**Important Note:** Software is required to implement a watch dog timer. If results are not reported within reasonable time (1sec could be an upper limit), software should re-trigger the PTM sequence.
	T4 – T1 overflow	Return to Init state + logged in STATUS and No report msg to MNG. Software should clear the STATUS and set the Single Step to start again.	

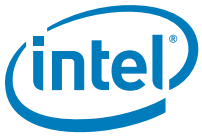


Table 3-8. PTM Exceptions

Test Condition	Exception	Current implementation	Comment
First Steps in a Cyclic setting	Time Out or PTM Res instead of PTM ResD	Return to Init state. No log in STATUS and No report msg to MNG. Auto restart the first step (no need to clear STATUS).	
	T4 – T1 overflow	Return to Init state + logged in STATUS reg and No report msg to MNG. The exception status indication propagates to the next cycle that will be completed OK. Auto restart the first step (no need to clear STATUS).	
N' Step in a Cyclic setting	Time Out	Return to Init state + Logged in STATUS + Report msg to MNG. Need to clear STATUS to start again.	
	T4 – T1 overflow or PTM Res instead of PTM ResD	Return to Init state + Logged in STATUS + Report msg to MNG. Auto restart the first step (no need to clear STATUS).	

3.1.4.6 Reporting Format of the PTM Results

Following a completion of the PTM cycle, Foxville reports the results in a receive packet that is posted to host memory as any other packet received from the network. The reported parameters are listed in the Table 3-9 below (byte offset 0x10 and on). In order to receive this reported packet, software should program one of the unicast MAC addresses to 0x000000000000 and assign a target receive queue in the RAL, RAH registers.

Table 3-9. PTM Message Format

Byte Offset	Field Size [Bytes]	Description	Value
0	6	Destination MAC address	0x000000000000
6	6	Source MAC Address	0x000000000000
0xC	2	EtherType Code	0x0001
0xE	2	Status (16 LS bits of PTM_STAT)	0x0
0x10	8	Timer 0 at T1' time [sec, nsec units]	Variable
0x18	8	Timer 1 at T1' time [sec, nsec units]	Variable
0x20	8	Timer 2 at T1' time [sec, nsec units]	Variable
0x28	8	Timer 3 at T1' time [sec, nsec units]	Variable
0x30	8	T2 [nsec]	Variable
0x38	4	T3 - T2 [nsec]	Variable
0x3C	4	T4' - T1' [nsec]	Variable
0x40	4	T delay [nsec] (valid only if the PTM_Prev_T3m2 is reported in little endian)	Variable
0x44	4	CRC Field	0x0

3.1.4.7 PTM Capability in the PCIe Configuration Space

See Section 9.5.5.

3.1.5 Transaction Layer

The upper layer of the PCIe architecture is the transaction layer. The transaction layer connects to Foxville core using an implementation specific protocol. Through this core-to-transaction-layer protocol, the application-specific parts of Foxville interact with the PCIe subsystem and transmit and receive requests to or from the remote PCIe agent, respectively.

3.1.5.1 Transaction Types Accepted by Foxville

Table 3-10. Transaction Types Accepted by the Transaction Layer

Transaction Type	FC Type	Tx Later 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	-	-
I/O Read Request	NPH	CPLH + CPLD	Requester ID, TAG, Attribute
I/O Write Request	NPH + NPD	CPLH	Requester ID, TAG, Attribute
Read Completions	CPLH + CPLD	-	-
Message	PH+ PD[1]	-	-

1. MCTP messages contains a payload.

Flow control types:

- PH - Posted request headers
- PD - Posted request data payload
- NPH - Non-posted request headers
- NPD - Non-posted request data payload
- CPLH - Completion headers
- CPLD - Completion data payload

3.1.5.1.1 Configuration Request Retry Status

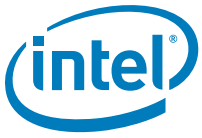
PCIe supports devices requiring a lengthy self-initialization sequence to complete before they are able to service configuration requests. This is the case for Foxville where initialization is long due to the NVM read operation following reset.

If the read of the PCIe section in the NVM was not completed and Foxville receives a configuration request, Foxville responds with a configuration request retry completion status to terminate the request. This effectively stalls the configuration request until the subsystem completes a local initialization and is ready to communicate with the host.

3.1.5.1.2 Partial Memory Read and Write Requests

Foxville has limited support of read and write requests when only part of the byte enable bits are set as described later in this section.

Partial writes to the MSI-X table are supported. All other partial writes are ignored and silently dropped.



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).

Partial reads with at least one byte enabled are answered as a full read. Any side effect of the full read (such as clear by read) is applicable to partial reads also.

Zero-length reads generate a completion, but the register is not accessed and undefined data is returned.

3.1.5.2 Transaction Types Initiated by Foxville

Table 3-11. Transaction Types Initiated by the Transaction Layer

Transaction type	Payload Size	FC Type	From Client
Configuration Read Request Completion	Dword	CPLH + CPLD	Configuration space
Configuration Write Request Completion	-	CPLH	Configuration space
I/O Read Request Completion	Dword	CPLH + CPLD	CSR
I/O Write Request Completion	-	CPLH	CSR
Read Request Completion	Dword/Qword	CPLH + CPLD	CSR
Memory Read Request	-	NPH	DMA
Memory Write Request	<= MAX_PAYLOAD_SIZE[1]	PH + PD	DMA
Message	64 bytes[2]	PH	INT / PM / Error Unit / LTR

1. MAX_PAYLOAD_SIZE supported is loaded from NVM (128 bytes, 256 bytes or 512 bytes). Effective MAX_PAYLOAD_SIZE is defined according to configuration space register.
2. MCTP messages contains payload.

3.1.5.2.1 Data Alignment

Requests must never specify an address/length combination that causes a memory space access to cross a 4 KB boundary. Foxville 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 limiting buffer sizes and base addresses to comply with a 4 KB boundary in cases where it improves performance.

The general rules for packet alignment are as follows:

1. The length of a single request does not exceed the PCIe limit of MAX_PAYLOAD_SIZE for write and MAX_READ_REQ for read.
2. The length of a single request does not exceed Foxville's internal limitation.
3. A single request does not span across different memory pages as noted by the 4 KB boundary previously mentioned.

Note: The rules apply to all Foxville requests (read/write, snoop and no snoop).

If a request can be sent as a single PCIe packet and still meet rules 1-3, then it is not broken at a cache-line boundary (as defined in the PCIe Cache Line Size configuration word), but rather, sent as a single packet (motivation is that the chipset might break the request along cache-line boundaries, but Foxville should still benefit from better PCIe use). However, if rules 1-3 require that the request is broken into two or more packets, then the request is broken at a cache-line boundary.

3.1.5.2.2 Multiple Tx Data Read Requests (MULR)

Foxville supports pipelined requests for transmit data on port. In general, the requests might belong to the same packet or to consecutive packets to be transmitted on the LAN port. However, the following restriction applies: all requests for a packet are issued before a request is issued for a consecutive packet.

Read requests can be issued from any of the supported queues, as long as the restriction is met. Pipelined requests might belong to the same queue or to separate queues. However, as previously noted, all requests for a certain packet are issued (from same queue) before a request is issued for a different packet (potentially from a different queue).

The PCIe specification does not ensure that completions for separate requests return in-order. Read completions for concurrent requests are not required to return in the order issued. Foxville handles completions that arrive in any order. Once all completions arrive for a given request, Foxville might issue the next pending read data request.

- Foxville incorporates a re-order buffer to support re-ordering of completions for all requests. Each request/completion can be up to 2 KB long. The maximum size of a read request is defined as the minimum {2 KB, Max_Read_Request_Size}.

In addition to the pipeline requests for transmit data, Foxville can issue up to read request to fetch transmit descriptors and read requests to fetch receive descriptors. The requests for transmit data, transmit descriptors, and receive descriptors are independently issued. Each descriptor read request can fetch up to 16 descriptors for reception and 24 descriptors for transmission.

3.1.5.3 Messages

3.1.5.3.1 Message Handling by Foxville (as a Receiver)

Message packets are special packets that carry a message code.

The downstream port of the system transmits special messages to Foxville by using this mechanism.

The transaction layer decodes the message code and responds to the message accordingly.

Table 3-12. Supported Message in Foxville (as a Receiver)

Message Code [7:0]	Routing r2r1r0	Message	Foxville 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
0x53	100b	PTM Response / PTM ResponD	Process the PTM cycle

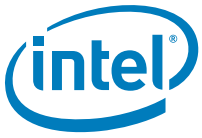


Table 3-12. Supported Message in Foxville (as a Receiver)

Message Code [7:0]	Routing r2r1r0	Message	Foxville Response
0x7E	000b 010b 011b 100b	Vendor_defined type 0	Drop and handle as an Unsupported Request
0x7F	100b	Vendor_defined type 1	Silently drop
0x7F	000b 010b 011b	Vendor_defined type 1	Send to MCTP reassembly if Vendor ID = 0x1AB4 (DMTF) and VDM code - 0000b (MCTP). Otherwise, silently drop

3.1.5.3.2 Message Handling by Foxville (as a Transmitter)

The transaction layer is also responsible for transmitting specific messages to report internal/external events (such as interrupts and PMEs).

Table 3-13. Supported Message in Foxville (as a Transmitter)

Message code [7:0]	Routing r2r1r0	Message
0x10	100	Latency Tolerance Reporting (LTR)
0x18	000	PM_PME
0x1B	101	PME_TO_ACK
0x20	100	Assert INT A
0x21	100	Not used
0x22	100	Not used
0x23	100	Not used
0x24	100	Deassert INT A
0x25	100	Not used
0x26	100	Not used
0x27	100	Not used
0x30	000	ERR_COR
0x31	000	ERR_NONFATAL
0x33	000	ERR_FATAL
0x52	100b	PTM Request (expected only to be transmitted from Foxville to the system)
0x7F	000, 010, 011,	VDM

3.1.5.4 Ordering Rules

Foxville meets the PCIe ordering rules (PCI-X rules) by following the PCI simple device model:

- Deadlock avoidance - Master and target accesses are independent. The response to a target access does not depend on the status of a master request to the bus. If master requests are blocked, such as due to no credits, target completions might still proceed (if credits are available).
- Descriptor/data ordering - Foxville does not proceed with some internal actions until respective data writes have ended on the PCIe link:

— Foxville does not update an internal header pointer until the descriptors that the header pointer relates to are written to the PCIe link.

— Foxville does not issue a descriptor write until the data that the descriptor relates to is written to the PCIe link.

Foxville might issue the following master read request from each of the following clients:

- Rx Descriptor Read
- Tx Descriptor Read
- Tx Data Read (up to)

Completing separate read requests are not guaranteed to return in order. Completions for a single read request are guaranteed to return in address order.

3.1.5.4.1 Out of Order Completion Handling

In a split transaction protocol, when using multiple read requests in a multi processor environment, there is a risk that completions arrive from the host memory out of order and interleaved. In this case, Foxville sorts the request completions and transfers them to the Ethernet in the correct order.

3.1.5.5 Transaction Definition and Attributes

3.1.5.5.1 Max Payload Size

Foxville policy to determine Max Payload Size (MPS) is as follows:

- Master requests initiated by Foxville (including completions) limits MPS to the value defined for the function issuing the request.
- Target write accesses to Foxville are accepted only with a size of one Dword or two Dwords. Write accesses in the range of (three Dwords, MPS, etc.) are flagged as UR. Write accesses above MPS are flagged as malformed.

Refer to Section 2.2.2 - TLPs with Data Payloads - Rules of the PCIe base specification.

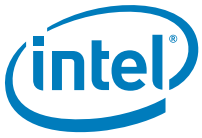
3.1.5.5.2 Traffic Class (TC) and Virtual Channels (VC)

3.1.5.5.3 Relaxed Ordering

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

- Relaxed ordering for descriptor and data reads: When Foxville emits 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.
- Relaxed ordering for receiving data writes: When Foxville issues receive DMA 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 complete.
- Foxville cannot relax ordering for descriptor writes, MSI/MSI-X writes or PCIe messages.

Relaxed ordering can be used in conjunction with the no-snoop attribute to enable the memory controller to advance non-snoop writes ahead of earlier snooped writes.



Relaxed ordering is enabled in Foxville by clearing the *RO_DIS* bit in the CTRL_EXT register.

3.1.5.5.4 Snoop Not Required

Foxville sets the *Snoop Not Required* attribute bit for master data writes. System logic might provide a separate path into system memory for non-coherent traffic. The non-coherent path to system memory provides higher, more uniform, bandwidth for write requests.

Note: The *Snoop Not Required* attribute does not alter transaction ordering. Therefore, to achieve maximum benefit from *Snoop Not Required* transactions, it is advisable to set the relaxed ordering attribute as well (assuming that system logic supports both attributes). In fact, some chipsets require that relaxed ordering is set for no-snoop to take effect.

Global no-snoop support is enabled in Foxville by clearing the *NS_DIS* bit in the CTRL_EXT register.

3.1.5.5.5 No Snoop and Relaxed Ordering for LAN Traffic

Software might configure non-snoop and relax order attributes for each queue and each type of transaction by setting the respective bits in the RXCTRL and TXCTRL registers.

Table 3-14 lists software configuration for the *No-Snoop* and *Relaxed Ordering* bits for LAN traffic when I/OAT 2 is enabled.

Table 3-14. LAN Traffic Attributes

Transaction	No-Snoop	Relaxed Ordering	Comments
Rx Descriptor Read	N	Y	
Rx Descriptor Write-Back	N	N	Relaxed ordering must never be used for this traffic.
Rx Data Write	Y	Y	Refer to Note 1 and Section 3.1.5.5.5.1
Rx Replicated Header	N	Y	
Tx Descriptor Read	N	Y	
Tx Descriptor Write-Back	N	Y	
Tx TSO Header Read	N	Y	
Tx Data Read	N	Y	

Note:
1. Rx payload no-snoop is also conditioned by the *NSE* bit in the receive descriptor. Refer to Section 3.1.5.5.5.1.

3.1.5.5.5.1 No-Snoop Option for Payload

Under certain conditions, which occur when I/OAT is enabled, software knows that it is safe to transfer (DMA) a new packet into a certain buffer without snooping on the front-side bus. This scenario typically occurs when software is posting a receive buffer to hardware that the CPU has not accessed since the last time it was owned by hardware. This might happen if the data was transferred to an application buffer by the I/OAT DMA engine.

In this case, software should be able to set a bit in the receive descriptor indicating that Foxville should perform a no-snoop DMA transfer when it eventually writes a packet to this buffer.

When a non-snoop transaction is activated, the TLP header has a non-snoop attribute in the *Transaction Descriptor* field.

This is triggered by the *NSE* bit in the receive descriptor. Refer to Section 7.1.4.2.

3.1.5.6 Flow Control

3.1.5.6.1 Foxville Flow Control Rules

Foxville implements only the default Virtual Channel (VC0). A single set of credits is maintained for VC0.

Table 3-15. Allocation of FC Credits

Credit Type	Operations	Number Of Credits
Posted Request Header (PH)	Target Write (one unit) Message (one unit)	Four units
Posted Request Data (PD)	Target Write (Length/16 bytes=1) Message (one unit)	MAX_PAYLOAD_SIZE/16
Non-Posted Request Header (NPH)	Target Read (one unit) Configuration Read (one unit) Configuration Write (one unit)	Four units
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:

- Foxville maintains four credits for NPD at any given time. It increments the credit by one after the credit is consumed and sends an UpdateFC packet as soon as possible. UpdateFC packets are scheduled immediately after a resource is available.
- Foxville provides four credits for PH (such as for four concurrent target writes) and four credits for NPH (such as for four concurrent target reads). UpdateFC packets are scheduled immediately after a resource becomes available.
- Foxville follows the PCIe recommendations for frequency of UpdateFC FCPs.

3.1.5.6.2 Upstream Flow Control Tracking

Foxville issues a master transaction only when the required FC credits are available. Credits are tracked for posted, non-posted, and completions (the later to operate with a switch).

3.1.5.6.3 Flow Control Update Frequency

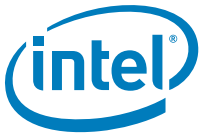
In any case, UpdateFC packets are scheduled immediately after a resource becomes available.

When the link is in the L0 or L0s link state, Update FCPs for each enabled type of non-infinite FC credit must be scheduled for transmission at least once every 30 µs (-0%/+50%), except when the *Extended Sync* bit of the Control Link register is set, in which case the limit is 120 µs (-0%/+50%).

3.1.5.6.4 Flow Control Timeout Mechanism

Foxville implements the optional FC update timeout mechanism.

The mechanism is activated when the link is in L0 or L0s Link state. It uses a timer with a limit of 200 µs (-0%/+50%), where the timer is reset by the receipt of any Init or Update FCP. Alternately, the timer can be reset by the receipt of any DLLP.



After timer expiration, the mechanism instructs the PHY to re-establish the link (via the LTSSM recovery state).

3.1.5.7 Error Forwarding

If a TLP is received with an error-forwarding trailer (poisoned TLP received), the transaction can either be resent or dropped and not delivered to its destination, depending on the *GCR.Completion Timeout resend enable* bit and the *GCR.Number of resends* field. If the re-sends were unsuccessful or if re-send is disabled, Foxville does not initiate any additional master requests for that PCI function until it detects an internal reset or a software reset for the LAN. Software is able to access device registers after such a fault.

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

3.1.6 Data Link Layer

3.1.6.1 ACK/NAK Scheme

Foxville sends an ACK/NAK immediately in the following cases:

1. NAK needs to be sent
2. ACK for duplicate packet
3. ACK/NAK before low power state entry

In all other cases, Foxville schedules an ACK transmission according to time-outs specified in the PCIe specification (depends on link speed, link width, and max_payload_size).

3.1.6.2 Supported DLLPs

The following DLLPs are supported by Foxville as a receiver:

Table 3-16. DLLPs Received by Foxville

DLLP type	Remarks
ACK	
NAK	
PM_Request_ACK	
InitFC1-P	Virtual Channel 0 only
InitFC1-NP	Virtual Channel 0 only
InitFC1-Cpl	Virtual Channel 0 only
InitFC2-P	Virtual Channel 0 only
InitFC2-NP	Virtual Channel 0 only
InitFC2-Cpl	Virtual Channel 0 only
UpdateFC-P	Virtual Channel 0 only
UpdateFC-NP	Virtual Channel 0 only
UpdateFC-Cpl	Virtual Channel 0 only

The following DLLPs are supported by Foxville as a transmitter:

Table 3-17. DLLPs Initiated by Foxville

DLLP type	Remarks
ACK	
NAK	
PM_Enter_L1	
PM_Enter_L23	
PM_Active_State_Request_L1	
InitFC1-P	Virtual Channel 0 only
InitFC1-NP	Virtual Channel 0 only
InitFC1-Cpl	Virtual Channel 0 only
InitFC2-P	Virtual Channel 0 only
InitFC2-NP	Virtual Channel 0 only
InitFC2-Cpl	Virtual Channel 0 only
UpdateFC-P	Virtual Channel 0 only
UpdateFC-NP	Virtual Channel 0 only

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

3.1.6.3 Transmit EDB Nullifying

If re-train is necessary, there is a need to guarantee that no abrupt termination of the Tx packet happens. For this reason, early termination of the transmitted packet is possible. This is done by appending an End Bad Symbol (EDB) to the packet.

3.1.7 Physical Layer

3.1.7.1 Link Speed

- Foxville supports 2.5GT/s / 5GT/s link speeds.

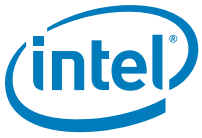
Foxville does not initiate a hardware autonomous speed change and as a result the *Hardware Autonomous Speed Disable* bit in the PCIe Link Control 2 register is hardwired to 0b.

Foxville 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 *Enter Compliance* bit in the PCIe Link Control 2 register.

3.1.7.2 Link Width

Foxville supports a maximum link width of x1 lane.

During link configuration, the platform and Foxville negotiate on a common link width. The link width must be x1.



3.1.7.3 Polarity Inversion

If polarity inversion is detected, the receiver must invert the received data.

During the training sequence, the receiver looks at Symbols 6-15 of TS1 and TS2 as the indicator of lane polarity inversion (D+ and D- are swapped). If lane polarity inversion occurs, the TS1 Symbols 6-15 received are D21.5 as opposed to the expected D10.2. Similarly, if lane polarity inversion occurs, Symbols 6-15 of the TS2 ordered set are D26.5 as opposed to the expected D5.2. This provides clear indication of lane polarity inversion.

3.1.7.4 Reset

The PCIe PHY can supply a core reset to Foxville. The reset can be caused by three sources:

1. Upstream move to hot reset - Inband Mechanism (LTSSM).
2. Recovery failure (LTSSM returns to detect).
3. Upstream component moves to disable.

3.1.7.5 Scrambler Disable

The scrambler/de-scrambler functionality in Foxville can be disabled by either one of the two connected devices according to the PCIe specification.

3.1.8 Error Events and Error Reporting

3.1.8.1 Mechanism in General

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 Foxville.

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

3.1.8.2 Error Events

Table 3-18 lists the error events identified by Foxville and the response in terms of logging, reporting, and actions taken. Consult the PCIe specification for the effect on the PCI Status register.

Table 3-18. Response and Reporting of PCIe Error Events

Error Name	Error Events	Default Severity	Action
PHY errors			
Receiver error	8b/10b decode errors Packet framing error	Correctable. Send ERR_CORR	TLP to initiate NAK and drop data. DLLP to drop.
Data link errors			

Table 3-18. Response and Reporting of PCIe Error Events (Continued)

Error Name	Error Events	Default Severity	Action
Bad TLP	• Bad CRC • Not legal EDB • Wrong sequence number	Correctable. Send ERR_CORR	TLP to initiate NAK and drop data.
Bad DLLP	• Bad CRC	Correctable. Send ERR_CORR	DLLP to drop.
Replay timer timeout	• REPLAY_TIMER expiration	Correctable. Send ERR_CORR	Follow LL rules.
REPLAY NUM rollover	• REPLAY NUM rollover	Correctable. Send ERR_CORR	Follow LL rules.
Data link layer protocol error	• Violations of Flow Control Initialization Protocol • Reception of NACK/ACK with no corresponding TLP	Uncorrectable. Send ERR_FATAL	Follow LL rules.
TLP errors			
Poisoned TLP received	• TLP with error forwarding	Uncorrectable. ERR_NONFATAL Log header	A poisoned completion is ignored and the request can be retried after timeout. If enabled, the error is reported.
Unsupported Request (UR)	• Wrong config access • MRdLk • Configuration request type 1 • Unsupported vendor Defined type 0 message • Not valid MSG code • Not supported TLP type • Wrong function number • Received TLP outside address range	Uncorrectable. ERR_NONFATAL Log header	Send completion with UR.
Completion timeout	• Completion timeout timer expired	Uncorrectable. ERR_NONFATAL	Error is non-fatal (default case): • Send error message if advisory • Retry the request once and send advisory error message on each failure • If fails, send uncorrectable error message Error is defined as fatal: • Send uncorrectable error message
Completer abort	• Received target access with data size > 64-bit	Uncorrectable. ERR_NONFATAL Log header	Send completion with CA.
Unexpected completion	• Received completion without a request for it (tag, ID, etc.)	Uncorrectable. ERR_NONFATAL Log header	Discard TLP.
Receiver overflow	• Received TLP beyond allocated credits	Uncorrectable. ERR_FATAL	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. Foxville doesn't report violations of flow control initialization protocol

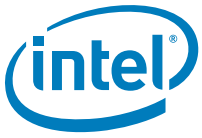


Table 3-18. Response and Reporting of PCIe Error Events (Continued)

Error Name	Error Events	Default Severity	Action
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 • Power management messages that doesn't use TC0. • Usage of unsupported VC.	Uncorrectable. ERR_FATAL Log header	Drop the packet and free FC credits.
Completion with unsuccessful completion status		No action (already done by originator of completion).	Free FC credits.
Byte count integrity in completion process.	When byte count isn't compatible with the length field and the actual expected completion length. For example, length field is 10 (in Dword), actual length is 40, but the byte count field that indicates how many bytes are still expected is smaller than 40, which is not reasonable.	No action	Foxville doesn't check for this error and accepts these packets. This might cause a completion timeout condition.

3.1.8.3 Error Forwarding (TLP Poisoning)

If a TLP is received with an error-forwarding trailer, the transaction can be re-sent a number of times as programmed in the GCR register. If transaction still fails the packet is dropped and is not delivered to its destination. Foxville then reacts as listed in Table 3-18.

Foxville does not initiate any additional master requests for that PCI function until it detects an internal software reset for LAN port. Software is able to access device registers after such a fault.

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 instead of the faulty area, etc.

3.1.8.4 ECRC

Foxville supports End to End CRC (ECRC) as defined in the PCIe specification. The following functionality is provided:

- Inserting an ECRC in all transmitted TLPs:
  - Foxville indicates support for inserting ECRC in the *ECRC Generation Capable* bit of the PCIe configuration registers. This bit is loaded from the ECRC Generation NVM bit.
  - Inserting an ECRC is enabled by the *ECRC Generation Enable* bit of the PCIe configuration registers. For MCTP packets, it is also controlled by the ECRC Generation for MCTP in PCIe Control 2 NVM word.
- 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), which would result in re-issuing the request.
  - Foxville indicates support for ECRC checking in the *ECRC Check Capable* bit of the PCIe configuration registers. This bit is loaded from the ECRC Check NVM bit.
  - ECRC checking is enabled by the *ECRC Check Enable* bit of the PCIe configuration registers.
- ECRC errors are reported.

3.1.8.5 Partial Read and Write Requests

3.1.8.5.1 Partial Memory Accesses

Foxville has limited support of read/write requests with only part of the byte enable bits set:

- Partial writes with at least one byte enabled should not be used. If used, the results are unexpected, either the byte enable request is honored or the entire Dword is written.
- Zero-length writes has no internal impact (nothing written, no effect such as clear-by-write). The transaction is treated as a successful operation (no error event).
- Partial reads with at least one byte enabled are handled as a full read. Any side effect of the full read (such as clear by read) is also applicable to partial reads.
- Zero-length reads generate a completion, but the register is not accessed and undefined data is returned.

Foxville does not generate an error indication in response to any of the above events.

3.1.8.5.2 Partial I/O Accesses

- Partial access on address
  - — A write access is discarded
  - — A read access returns 0xFFFF
- Partial access on data, where the address access was correct
  - — A write access is discarded
  - — A read access performs the read

3.1.8.6 Error Pollution

Error pollution can occur if error conditions for a given transaction are not isolated on the error's first occurrence. If the physical layer detects and reports a receiver error, to avoid having this error propagate and cause subsequent errors at 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.

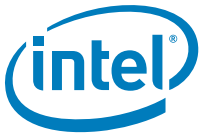
3.1.8.7 Completion with Unsuccessful Completion Status

A completion with unsuccessful completion status is dropped and not delivered to its destination. An interrupt is generated to indicate unsuccessful completion.

3.1.8.8 Error Reporting Changes

The Rev. 1.1 specification defines two changes to advanced error reporting. A new *Role-Based Error Reporting* bit in the Device Capabilities register is set to 1b to indicate that these changes are supported by Foxville. These changes are:

1. Setting the *SERR# Enable* bit in the PCI Command register also enables UR reporting (in the same manner that the *SERR# Enable* bit enables reporting of correctable and uncorrectable errors). In other words, the *SERR# Enable* bit overrides the UR *Error Reporting Enable* bit in the PCIe Device Control register.



2. Changes in the response to some uncorrectable non-fatal errors, detected in non-posted requests to Foxville. These are called advisory non-fatal error cases. For each of the errors that follow, the following behavior is defined:
   a. The *Advisory Non-Fatal Error Status* bit is set in the Correctable Error Status register to indicate the occurrence of the advisory error and the *Advisory Non-Fatal Error Mask* corresponding bit in the Correctable Error Mask register is checked to determine whether to proceed further with logging and signaling.
   b. If the *Advisory Non-Fatal Error Mask* bit is clear, logging proceeds by setting the corresponding bit in the Uncorrectable Error Status register, based upon the specific uncorrectable error that's being reported as an advisory error. If the corresponding uncorrectable error bit in the Uncorrectable Error Mask register is clear, the First Error Pointer and Header Log registers are updated to log the error, assuming they are not still occupied by a previously unserviced error.
   c. An ERR_COR message is sent if the *Correctable Error Reporting Enable* bit is set in the Device Control register. An ERROR_NONFATAL message is not sent for this error.

The following uncorrectable non-fatal errors are considered as advisory non-fatal Errors:

- A completion with an Unsupported Request or Completer Abort (UR/CA) status that signals an uncorrectable error for a non-posted request. If the severity of the UR/CA error is non-fatal, the completer must handle this case as an advisory non-fatal error.
- When the requester of a non-posted request times out while waiting for the associated completion, the requester is permitted to attempt to recover from the error by issuing a separate subsequent request, or to signal the error without attempting recovery. The requester is permitted to attempt recovery zero, one, or multiple (finite) times, but must signal the error (if enabled) with an uncorrectable error message if no further recovery attempts are made. If the severity of the completion timeout is non-fatal and the requester elects to attempt recovery by issuing a new request, the requester must first handle the current error case as an advisory non-fatal error.
- Reception of a poisoned TLP. Refer to Section 3.1.8.3.
- When a receiver receives an unexpected completion and the severity of the unexpected completion error is non-fatal, the receiver must handle this case as an advisory non-fatal error.

3.1.8.9 Completion with Unsupported Request (UR) or Completer Abort (CA)

A DMA master transaction ending with an Unsupported Request (UR) completion or a Completer Abort (CA) completion causes all PCIe master transactions to stop, *PICAUSE.ABR* bit is set and an interrupt is generated if the appropriate *Mask* bits are set. To enable PCIe master transactions after receiving an UR or CA completion, software should issue a Device Reset (*CTRL.DEV_RST*) and re-initialize the function.

Note: Asserting *CTRL.DEV_RST* flushes any pending transactions on the PCIe and reset's port.

3.1.9 Performance Monitoring

Foxville incorporates PCIe performance monitoring counters to provide common capabilities to evaluate performance. Foxville implements four 32-bit counters to correlate between concurrent measurements of events as well as the sample delay and interval timers. The four 32-bit counters can also operate in a two 64-bit mode to count long intervals or payloads. software can reset, stop, or start the counters (all at the same time).

The list of events supported by Foxville and the counters control bits are described in the memory register map (Section 8.6).

Some counters operate with a threshold - the counter is incremented only when the monitored event crossed a configurable threshold (such as the number of available credits is below a threshold).

The list of events supported by Foxville and the counters Control bits are described in the PCIe Register section.

3.1.10 PCIe Power Management

Described in Section 5.4.1 - Power Management.

3.1.11 PCIe Programming Interface

Described in Chapter 9.0 - PCIe Programming Interface

3.2 Management Interfaces

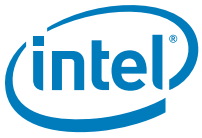
Foxville contains two possible interfaces to an external MC (CSME). SMBus and PCIe. See Section 10.1 for more details on the management interfaces.

3.3 Non-Volatile Memory (NVM) Flash

3.3.1 General Overview

Foxville uses a Flash device for storing product configuration information. The Flash is consist of several "blocks" listed below, summarized in Table 3-19 and illustrated in Figure 3-5. More granular modules in the NVM are listed in Table 3-19. The various blocks and its programming method are detailed in the following sections. For its operation Foxville requires a Flash of 1M Byte as a minimum and 2M Byte if OROM is needed.

- **Legacy EEPROM** - Hardware Accessed Modules (**Block 0** and **Block 1**) — Loaded by the Foxville hardware after power-up, PCI reset de-assertion, D3 to D0 transition, or software reset. Different hardware sections in the Flash are loaded at different events. For more details on power-up and reset sequences, see Section 4.0. The hardware "block" is loaded to a shadow RAM in the device. It is located at the first 2 x 4KB sectors in the Flash. See description of this "block" and its programming in Section 3.3.2.
- **Secured Image** (**Block 2**) — The following data sections are protected by an RSA signature:
  - **CSS Header** - The CSS header describes the modules within the protected block and includes the security signature of the block. See Section 3.3.10.2 for a description of the CSS header. The CSS header is counted as part of the FW module for the sake of its size in the NVM.
  - **FW Module** - The Firmware module contains the code for the embedded controller, a list of supported Flash devices and a RO update section. The Firmware module is loaded at power up and FW reset.
  - **PHY Module** - The PHY code is loaded by the PHY cluster for its functionality following a power up or any reset that affect the PHY.
  - **Expansion/Option ROM Module** (OROM Module) - This block is optional and might exist only on larger FLASH sizes than 1MB. It holds a code expected to be executed by the system at pre-OS state. The option ROM module may include the following sub-modules: PXE driver, iSCSI boot image, UEFI network driver and can also include a CLP module.
- **Free Provisioning Module** (**Block 3**) — This block is used by the software to program a new Secured image. The programming flow is described in Section 3.3.10.1.
- **mDNS Records A / B** (**Block3 / Block 4**) — The mDNS records are used by the CSME firmware for mDNS offload proxy while the system is in D3. It is the software responsibility to program these records before entering the D3 state. The mDNS records are expected to reside adjacent to the



Firmware space. So, the mDNS records are stored in the "A" space or "B" space (as shown in Figure 3-5) depending in the location of the secured image.

- **Software Free Access Module** (**Block 4**) — This module is used mainly by the software. It also contain the mDNS records used by the ME firmware for mDNS offload proxy while the system is in D3. The structure of this block is outside the scope of this document.

Table 3-19. NVM Modules

Module Name	Module Size	Module Pointer Location
Shadow RAM Sector 0	4 KB	Module starts at absolute address 0 in the Flash
Shadow RAM Sector 1	4 KB	Module starts at absolute address 4K Byte in the Flash
CSS header	inc. in FW Module	Word offset 0x10 in the shadow RAM.
FW Module	364 KB	The address is defined in the CSS header.
PHY Module	128 KB	Word offset 0x7F0 in the shadow RAM.
OROM Module	512 KB	Offset 0x4A in the shadow RAM.
Free Provisioning Space	1004 KB	Offset 0x40 in the shadow RAM.
Software Free	16 KB	Word offset 0x7F1 in the shadow RAM.
mDNS Records	16 KB	Offset 0x25 in the shadow RAM.
All above pointers are 16 bit words. The MS bit of these pointers are set to '1' stating that the pointers define an absolute address outside the shadow RAM space. The other 15 LS bits of these pointers specify an address defined in 4KB units.		

Table 3-20. NVM Module Names

Module Name	NVM Section	Module Pointer	Write Protected
Shadow RAM Sectors 0, 1 at addresses 0x0000 and 0x0800.	Section 6.1.1	Module starts at absolute address 0 , 4KB in the Flash	Specific Words
PXE VLAN Configuration	Section 6.1.4	Address is defined by the "PXE VLAN pointer" in the NVM	N
Alternate MAC Address Section	Section 6.1.12	Address is defined by the "Alternate MAC Address Location" in the NVM	N
RO start section	Section 6.1.5	Defined by the "Start of RO Area" word in the NVM	Y
LPG Reset CSR Auto Configuration	Section 6.1.6	Address is defined by the "CSR Auto Configuration Power-Up Pointer" in the NVM	**Y**
Firmware Module Configuration	Section 6.1.2	Module address is defined by the "Firmware Module Configuration Pointer" in word offset 0x0051	**Y**
Common Firmware Parameters	Section 6.1.3	Address is defined by the "Common Firmware Parameters pointer" in the "Firmware Module"	**Y**
SideBand Configuration Section	Section 6.1.8	Address is defined by the "SideBand Configuration Pointer" in the FW module in the NVM	**Y**
Flexible TCO Filter Configuration Section	Section 6.1.9	Address is defined by the "Flexible TCO Filter Configuration Pointer" in the FW module in the NVM	**Y**
Traffic Types Parameters Section	Section 6.1.10	Address is defined by the "Traffic Types Parameters Pointer" in the FW module in the NVM	**Y**
PBA Section	Section 6.1.13	Internal structure of the FW image	~~obsolete~~Y
VPD Module Section	Section 6.1.15	Address is defined by the "VPD Pointer" in the NVM	Y
RO end section Section	Section 6.1.7	Defined by the "End of RO Area" word in the NVM	Y
PHY Configuration Section	Section 6.1.11	~~Defined by the "Pointer - GPHY SW Section" at 0x7F0~~	obsolete
iSCSI Boot Configuration Section	Section 6.1.15	Address is defined by the "iSCSI Boot Configuration Pointer" word in the NVM	N

Table 3-20. NVM Module Names

Module Name	NVM Section	Module Pointer	Write Protected
Pointer - GPHY FW Section	Section 6.1.16	Word offset 0x07F0	**Y**
Pointer - SW Free Space Section	Section 6.1.17	Word offset 0x07F1	**Y**
EXP. ROM Boot Code	Section 6.1.18	Address is defined by the "EXP. ROM Boot Code Section Pointer" word in the NVM at address 0x7F0	Y
GPHY Firmware Section	Section 6.1.19	Address is defined by the "Pointer - GPHY FW Section" word in the NVM	Y
CSS header Section	Section 6.1.20	Internal structure of the FW image	Y
FW_HEADER in REGMAN	Section 6.1.21	Internal structure of the FW image	Y
FW Image Section	Section 6.1.22	Address is defined by the "Firmware Image Pointer" word in the NVM	Y
Flash Info Section	Section 6.1.23	Internal structure of the FW image	Y
RO Updates Info Section	Section 6.1.24	Internal structure of the FW image	Y
CSS-signed Firmware Secured Module	Section 6.1.25	Internal structure of the FW image	Y
Free Provisioning Area Section	Section 6.1.26	Address is defined by the "Free Provisioning Area Pointer" word in the NVM	N
mDNS Records Section	Section 6.1.27	Address is defined by the "mDNS Records Area Offset" word in the NVM	N
SW Free Space Section	Section 6.1.28	Address is defined by the "Pointer - SW Free Space Section" word in the NVM	N

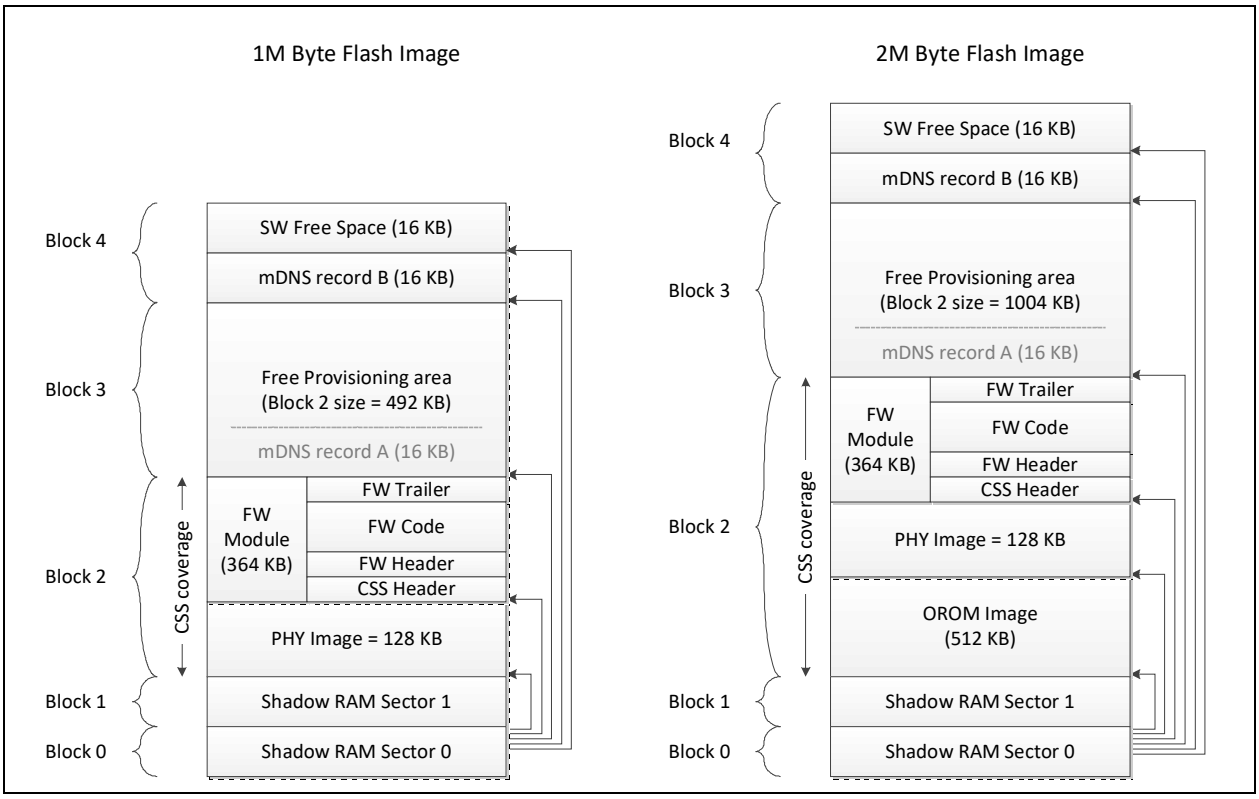


Figure 3-5. NVM Structure

Note: Along this document the terms NVM and Flash are used to refer to the non-volatile memory used by Foxville. both terms have the same meaning.

Foxville merges the "legacy EEPROM" and Flash content in a single Flash device. Flash devices require a sector erase instruction in case 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. Foxville supports Flash devices with a sector erase size of 4 KB. Note that many Flash vendors are using the term sector differently. Foxville EAS uses the term Flash sector for a logic section of 4 KB.

Foxville supports Flash devices that are either write-protected by default after power-up or not. Foxville FW RAM-based code removes the protection by sending the write-protection removal OpCode to the Flash after power up. For the first programming of a blank Flash, it is the host's software tool responsibility to remove the write-protection from the Flash part via bit-banging access.

The following OpCodes are supported by Foxville as they are common to all the supported Flash devices:

CMD	Opcode	CMD Description
WREN	0x06	Write Enable
RDID	0x9F	Read JEDEC Manufacture ID
WRSR	0x01	Write Status

CMD	Opcode	CMD Description
READ	0x03	Flash Read
RDSR	0x05	Read Flash Status
SERASE	0x20	4 KB Sector Erase
FERASE	0xC7	Flash Erase
Page Program	0x02	Write to the Flash
Fast READ	0x0B	Read data bytes at higher speed
Other OpCodes to be supported are loaded from the firmware secured area into a set of Flash Opcode registers		

3.3.1.1 Flash Detection, NVM Validity Field, and Non-Secure Mode

Foxville supports detection of Flash existence following power-up and detection of a valid NVM image. Foxville enters non-secure mode (blank Flash programming mode) in the following scenarios (listed in the table below). In the rest of this document, the NVM is assumed to be secured unless specified otherwise.

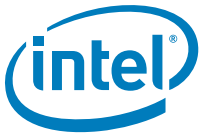
Setting Options	Device Functionality
No Flash device is detected	Auto-load from Flash by hardware or firmware after power-up or reset is not performed. The internal firmware is not enabled as well.
Invalid Signature at NVM word 0x12	
Incorrect checksum at NVM word 0x3F	
The NVM_SEC_EN = 0b at NVM word 0x12	Reflected in the BLOCK_PROTECTED_SW_ACCESS flag in the FL_SECU register. The internal FW is enabled and the whole Flash image is unprotected. It can be programmed by the "Software Flash" registers or bit banging by the FLA register.
Device ID field read from firmware image in the NVM is not Foxville's Device ID	The internal FW is not functional and the whole Flash image is unprotected.

3.3.2 Shadow RAM

Foxville maintains the first two 4 KB sectors, Sector 0 and Sector 1 (block 0 and block 1), for the hardware configuration content. At least one of these two sectors must be valid at any given time or else Foxville is set by hardware default. Following a Power On Reset (POR), Foxville copies the valid lower 4 KB sector of the Flash device into an internal shadow RAM. Any further accesses of the software or firmware to this section of the NVM are directed to the internal shadow RAM. After a software command, modifications made to the shadow RAM content are then copied by Foxville manageability into the other 4 KB sector of the NVM, flipping circularly the valid sector between sector 0 and 1 of NVM.

This mechanism provides the following advantages:

1. A seamless backward compatible read/write interface for software/firmware to the first 4 KB of the NVM as if an external EEPROM device was connected. This interface is referred as EEPROM-mode access to the Flash.
2. A way for software to protect image-update procedure from power down events by establishing a double-image policy. It relies on having pointers to all the other NVM modules mapped in the NVM sector which is mirrored in the internal shadow RAM.



3. A way to ensure that a hardware auto-load event, which occurs further to a PCIe reset event, can be completed within the PCIe timing constraints (100 ms) even if the Flash is occupied performing an erase operation initiated just before the reset.

Due to NVM security reasons, hardware does not allow any Flash accesses until the NVM is authenticated and the Flash blocks are identified by the device. See more on NVM security in Section 3.3.11.

Following a write access by software or firmware to the shadow RAM, the data should finally be updated in the Flash as well. Foxville manageability updates the Flash from the shadow RAM when software requests explicitly to update the Flash by setting the *FLUPD* bit in the EEC register. For saving Flash updates, it is expected that software set the *FLUPD* bit only once it has completed the last write access to the Flash. Foxville manageability then copies the content of the shadow RAM to the non-valid configuration sector and makes it the valid one.

Notes: Software should be aware that programming the Flash might require a long latency due to the Flash update sequence handled by manageability. The sector erase command by itself can last hundreds of milliseconds. Software must poll the *FLUDONE* bit in the EEC register to check whether or not the Flash programming completed.

Each time the Flash content is not valid (blank configuration sectors or wrong NVM Validity field contents in both sector 0 and 1) EEPROM access mode is turned off. Software should rather use one of the three flash access means described in Section 3.3.3.

Hardware auto-load process that occurs further to a reset event (other than power-up) is performed from internal shadow RAM and not from the Flash device.

3.3.2.1 Initialization from the Shadow RAM

Foxville initializes some of its registers from the Shadow RAM as indicated in the Table 3-21 below. All registers indicated in the table are initialized at power on reset and some of them are loaded following a software reset (caused by CTRL.DEV_RST bit or the CTRL_EXT.EE_RST):

Table 3-21. Shadow RAM Auto-load words

Word Offset	Word Name	Init Order	PCIe Reset	SW Reset	Word Offset	Word Name	Init Order	PCIe Reset	SW Reset
0x00:0x02	Ethernet MAC Address	47	Y	Y	0x24	Initialization Control 3	41	Y	
0x0A	Initialization Control Word 1	4	Y		0x27	CSR Auto Configuration Power-Up Pointer	3		
0x0B	Subsystem ID	35	Y		0x28	PCIe Control 2	9	Y	
0x0C	Subsystem Vendor ID	36	Y		0x29	PCIe Control 3	10	Y	
0x0D	Device ID	37	Y		0x2A	CDQM Memory Base Low	29	Y	
0x0E	Vendor ID	38	Y		0x2B	CDQM Memory Base High	30	Y	
0x0F	Initialization Control Word 2	46	Y	Y	0x2E	Watchdog Configuration	50	Y	Y
0x11	Flash Device Size	2			0x34	PCIe PHY Configuration 0 Low	11	Y	
0x12	EEPROM Sizing and Protected Fields	1	Y	Y	0x35	PCIe PHY Configuration 0 High	12	Y	
0x13	Initialization Control 4	45	Y	Y	0x38	PCIe PHY Configuration 1 Low	13	Y	
0x14	PCIe L1 Exit latencies	33	Y		0x39	PCIe PHY Configuration 1 High	14	Y	
0x15	PCIe Completion Timeout Configuration	34	Y		0x3A	PCIe PHY Configuration 2	15	Y	
0x16	MSI-X Configuration	39	Y		0x44	PCIe L1 Substates Cap Low	16	Y	

Table 3-21. Shadow RAM Auto-load words

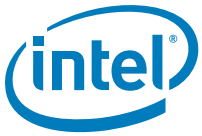
Word Offset	Word Name	Init Order	PCIe Reset	SW Reset	Word Offset	Word Name	Init Order	PCIe Reset	SW Reset
0x17	SW Reset CSR Auto Configuration Pointer	49	Y	Y	0x45	PCIe L1 Substates Cap High	17	Y	
0x18	PCIe Init Configuration 1	5	Y		0x46	PCIe L1 Substates Control 1st Low	18	Y	
0x19	PCIe Init Configuration 2	6	Y		0x47	PCIe L1 Substates Control 1st High	19	Y	
0x1A	PCIe Init Configuration 3	7	Y		0x48	PCIe L1 Substates Control 2nd	20	Y	
0x1B	PCIe Control 1	8	Y		0x49	PCIe PTM Control	21	Y	
0x1C	LED 1,3 Configuration	42	Y		0x4A	EXP. ROM Boot Pointer	22	Y	
0x1E	Device Rev ID	32	Y		0x52	PCIe PHY Configuration 3 Low	23	Y	
0x1F	LED 0,2 Configuration	43	Y		0x53	PCIe PHY Configuration 3 High	24	Y	
0x20	Software Defined Pins Control	48	Y	Y	0x54	PCIe PHY Configuration 4 Low	25	Y	
0x21	Functions Control	31	Y		0x55	PCIe PHY Configuration 4 High	26	Y	
0x22	LAN Power Consumption	40	Y		0x56	PCIe PHY Configuration 5 Low	27	Y	
0x23	PCIe Reset CSR Auto Configuration Pointer	44	Y		0x57	PCIe PHY Configuration 5 High	28	Y	

3.3.3 NVM Clients and Interfaces

There are several clients that can access the NVM to different address ranges via different access modes, methods, and interfaces. The various clients to the NVM are Hardware, software tools (BIOS, etc.), drivers, Firmware, BMC (via firmware), and VPD software. The Table 3-22 below lists the different accesses to the NVM.

Table 3-22. Clients and Access Types to the NVM

Client	NVM Access Method	NVM Access Mode	Logical Byte Address Range	NVM Access Interface (CSRs or Other)
VPD Software	Parallel (32-bits)	EEPROM	0x000000 - 0x0003FF	VPD Address and Data registers (PCI_E config space), via shadow RAM logic. Any write access is pushed by Foxville into the Flash as soon as possible. VPD module must be located in the first valid Flash sector.



Client	NVM Access Method	NVM Access Mode	Logical Byte Address Range	NVM Access Interface (CSRs or Other)
Software	Parallel (16-bits)	EEPROM	0x000000 - 0x000FFF	EERD, EEWR, via shadow RAM logic.
	Parallel (32-bits)	Flash	0x000000 - 0x001FFF	Memory mapped via BARs. Write access to this range is not allowed when in Secure mode. The transaction is completed but not executed.
		Flash	0x002000 - 0xFFFFFF	Memory mapped via BARs.
	Parallel (32-bits)	Flash	0x000000 - 0x001FFF	FLSW* register set - Software/Flash burst control. Write access to this range is not allowed when in Secure mode. The transaction is completed with FLSWCTL.CMDV bit cleared and not executed.
		Flash	0x002000 - 0xFFFFFF	FLSW* register set - Software/Flash burst control
Software or Firmware	Bit-banging (1-bit)	Flash	0x000000 - 0xFFFFFFF	FLA. Access allowed to Software only when in the non-secure mode. Firmware access via FLA register might cause a firmware hang because cache read is not possible until the FLA access completes.
Firmware	Parallel (16-bits)	EEPROM	0x000000 - 0x000FFF	
Firmware	Parallel (32-bits)	Flash	0x000000 - 0xFFFFFF	FLMNG* register set.
Hardware	Serial pins	Flash	0x000000 - 0xFFFFFF	Write accesses are performed by the hardware on behalf of SW/FW entities. Read accesses are initiated by hardware at POR for loading the shadow RAM from flash or for the firmware cache reads or loading the PHY Firmware.
Hardware	Internal signals	EEPROM	0x000000 - 0x000FFF	Write accesses are performed by the hardware on behalf of SW/FW entities. Read accesses are initiated by hardware only for setting registers and memory defaults from shadow RAM (HW auto-load process), following reset events.

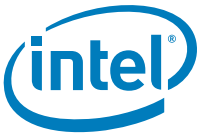
Note: Firmware saves words like SMBus Slave Addresses or Signature, which are saved into the NVM at the firmware's initiative. Software attempts to write access protected areas or words are silently discarded (access completed but not executed).

3.3.3.1 Memory Mapped Host Interface

The Flash device can be mapped into memory space through the use of Base Address Registers (BARs) and Flash control registers. Figure 3-6 show the BAR and control registers mapping to the FLASH device.

Clearing the *FLBAR_Size* and *CSR_Size* fields in PCIe Control 2 NVM word (Word 0x28) to 0b, disables Flash mapping to PCI space via the Flash Base Address register.

Setting the *LAN Boot Disable* bit in the Initialization Control 3 NVM word, disables Flash mapping to the PCI space via the Expansion ROM Base Address register



Using the legacy Flash transactions, the Flash is read from, or written to (under NVM security rules), by Foxville 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:

- The Expansion ROM Base Address Register (PCIe Control Register at offset 0x30). Host accesses to the Flash via the Expansion-ROM BAR are directed to the Flash at offsets relative to the "EXP. ROM Boot Code" pointer (see Figure 3-6). The "EXP. ROM Boot Code" pointer is located at word 0x004A in the Flash. If there is no valid NVM, then the Expansion ROM BAR is disabled.

For accesses through the BAR, the following occurs:

- If the Flash part is larger than the exposed BAR size (for saving operating system address space), accesses to the upper Flash addresses are not possible through the BAR.
- If the Flash part is smaller than the exposed BAR size (further to a wrong NVM setting or because of the 128 KB added for CSRs), accesses are (naturally) wrapped around when attempting to access upper addresses.

Foxville controls accesses to the Flash when it decodes a valid access. Attempts to out-of-range write access the PCIe Expansion/Option ROM module (beyond the provisioned 512 KB) is silently ignored, while read access might return any value. The same is done for out-of-range accesses to the host memory BAR.

Notes: Foxville supports four byte writes to the Flash. Byte Enable (BE) pins can be set in a consecutive way (starting from 0) for writing less than four bytes.

Flash read accesses are assembled by Foxville each time the access is greater than a byte-wide access.

Flash read access times is in the order of 2 µs (depending on Flash specification). The device continues to issue retry accesses during this time.

Flash write access times can be in the order of 2 µs to 200 µs (depending on Flash specification). Following a write access to the Flash, software should avoid initiating any read or write access to the device until the Flash write access completes.

Caution: While in the non-secure mode, Flash BAR access while FLA.FL_REQ is asserted (and granted) is forbidden. It can lead to a PCIe hang as a bit-banging access requires several PCIe accesses.

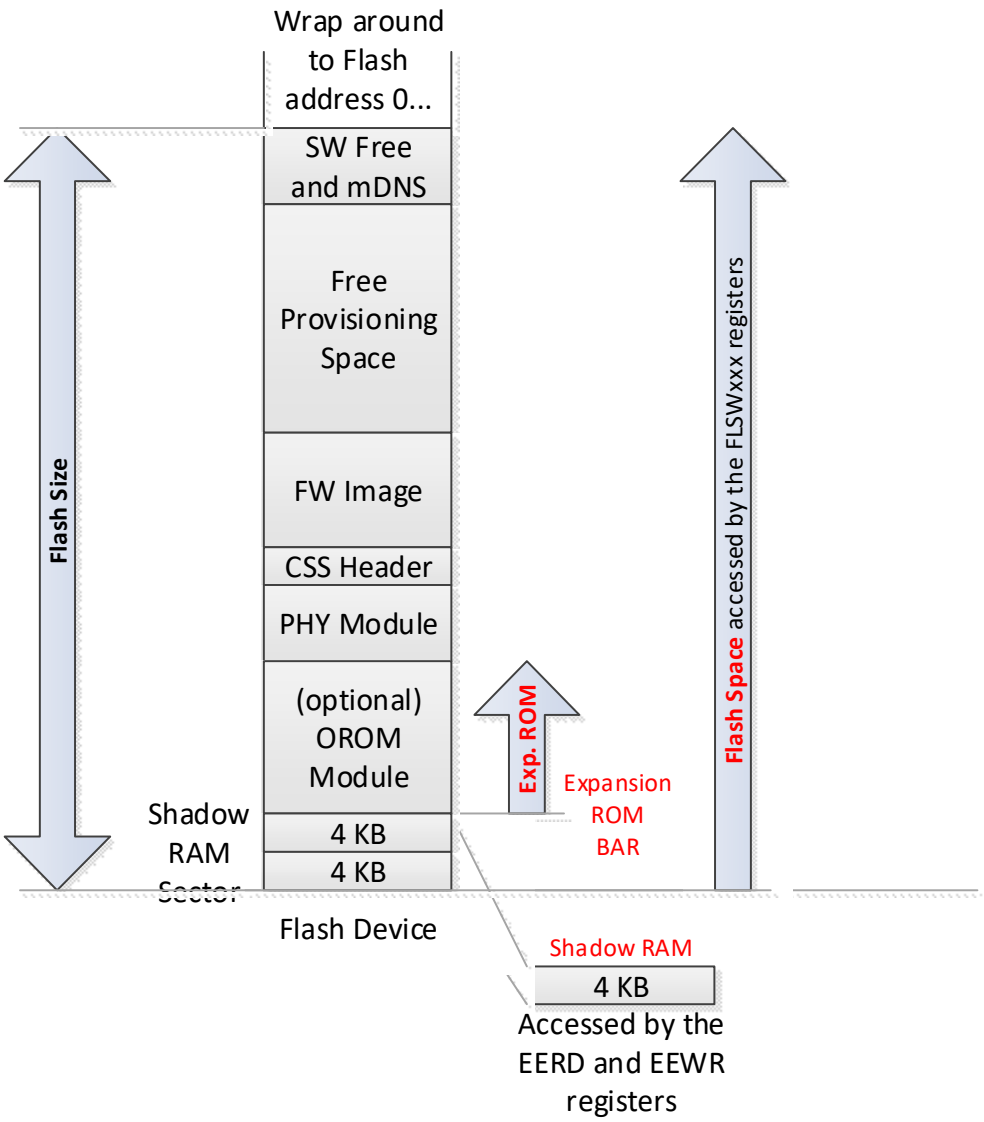


Figure 3-6. Flash Part versus the CSR and Expansion ROM BARs

3.3.3.2 Management Controller (MC) Interface

The MC can issue NC-SI commands that require read/write access to some Flash and/or shadow RAM words. Firmware is responsible to convert the NC-SI commands into EEPROM-mode or Flash-mode accesses to the required NVM locations.

3.3.4 Flash Access Contention

Foxville firmware is constantly running from the Flash. Any read or write access to the NVM made by software must be preceded by acquiring ownership over the NVM semaphore (refer to Section 4.8.1). 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.

Two software entities cannot use the semaphore mechanism: BIOS and VPD software.

- Since VPD software accesses only the VPD module, which is located in the first valid sector of the NVM, VPD accesses are always performed against the shadow RAM first. In this case, firmware must take/release ownership over the NVM before dumping the VPD changes into the Flash, as if it was the originator of the Flash access. Shadow RAM dump sequence is described in Section 3.3.2.
- No contention can occur between BIOS and any other software entity (VPD included) as it accesses the NVM while the operating system is down.
- Contention between BIOS and firmware can however happen if a system reboot occurs while the MC is accessing the NVM.
  - If a system reboot is caused by a user pressing the Standby button, it is required to route the wake-up signal from the Standby button to the MC and not to the chipset. The MC issues a system reboot signal to the chipset only after the NVM write access completes. Firmware is responsible to poll whether the NVM write has completed before sending the response to the MC NC-SI command.
  - If a system reboot is issued by a local user on the host, there is no technical way to avoid NVM access contention between BIOS and the MC.

Caution: It is the user's responsibility when accessing the NVM remotely via the MC to make sure another user is not currently initiating a local host reboot there.

Notes:

The MAC auto-load from the Flash device itself occurs only after power-up and before host or firmware can attempt to access the Flash. The host must wait until PCIe reset is de-asserted (after ~1 sec, which is enough time for the MAC auto-load to complete).

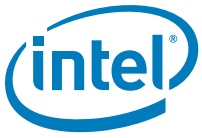
Software and firmware should avoid holding Flash ownership (via the dedicated semaphore bit) for more than 2 seconds.

Software erase command can be suspended by firmware until it handles its current tasks and/or it loads its cache.

3.3.4.1 Arbitration Between NVM Clients

The following lists the relative priority by which the hardware must serve the different NVM clients, whether the access is performed against the internal shadow RAM or into the Flash device:

1. **Hardware auto-load** - no semaphore taking.
   Pointer to an hardware module must first be invalidated (set to 0xFFFF) by the host/MC before modifying the module by a sequence of related changes. A sequence of related changes is a sequence of NVM writes that if interrupted in the middle would leave the hardware module with non-consistent contents. There is still a risk of inconsistent NVM header words being loaded by hardware if the auto-load process occurs in the middle of a write sequence performed over the NVM header. This risk exists in all previous 1 GbE controllers.
2. **Hardware cache read** for the firmware - no semaphore taking.
   This access is served by hardware with a 10:1 ratio relatively to the accesses that follow.
   The tens of cache read accesses (8-byte Flash read each) that might be performed before a memory mapped NVM access is served (see Steps 4 and 6) should not cause a timeout of the PCIe transaction.



Cache reads can be delayed by the maximum time duration (300 ms) of a previous erase command, which was issued by firmware to hardware. Before issuing any sector erase command to hardware, firmware must complete all its pending tasks and must load from Flash the code pieces required to manage while the Flash is busy for erasing:

a. For instance, NC-SI commands received are completed with the Package Not Ready status. The MC must retry after 500 ms the commands that were completed with a Package Not Ready status. SMBus transactions are handled in a similar way.

b. Offloads performed by firmware waits until the cache read resumes before being handled by firmware. Alternatively, before issuing an Erase command to hardware, firmware initiates some offload tasks that would timeout otherwise.

c. Host interface commands can wait for 500 ms before being completed by firmware.

3. **Software or firmware read** is performed by hardware in a round robin manner between:

   a. Software reads via BAR or CSRs - Software other than BIOS must take a semaphore (even for reads).

   b. Firmware reads via CSRs - no semaphore taking. Firmware is also performing the shadow RAM reads required for a VPD read.

4. **Firmware erase/write for VPD and the MC**, before performing a shadow RAM dump into the Flash, or for its own needs (such as for replacing factory defaults), firmware must take the semaphore here.

5. **Software erase/write via BAR or CSRs** - Software must take a semaphore.

   a. If the access is performed against the Flash, software must release the semaphore after it has checked the Flash is not busy by the last erase operation performed.

   b. If the access is performed against the shadow RAM, software must release the semaphore once it has asked firmware to dump the shadow RAM in the Flash by setting the *FLUPD* bit.

6. **Bit-banging access** - no semaphore taking.
   This access is provided to software only when in non-secure mode.Firmware access via the bit-banging interface might lead to a dead lock if the firmware code required to complete the bit-banging is not entirely in the firmware cache before starting the access.

3.3.5 NVM Read, Write, and Erase Sequences

This section describes the low-level NVM procedures handled between software, firmware, and hardware. The high-level NVM flows are built using these procedures, and include the semaphore taking/releasing and other high level tasks. Refer to Section 3.3.12.

Note: Each time programming the NVM via CSR accesses, PCIEMISC.DMA Idle Indication bit must be set to 1b.

3.3.5.1 Flash Erase Flow by Software or Firmware

In this section, software uses FLSW* registers, while firmware uses FLMNG* instead.

Note: Software may erase the Flash (using the FLSW* registers) only if the Flash is set to non-secured mode. If the software attempts to erase the whole Flash while in secured mode (nominal operation), the device sets the DONE bit without executing the command. Setting the DONE bit, an interrupt is issued to firmware.

Device Erase Flow:

1. Poll the FLSWCTL.DONE bit until it is set.
2. Set FLSWCTL.CMD fields to 0011b.
3. Wait until FLSWCTL.DONE bit is read as 1b and FLSWCTL.FLBUSY bit is read as 0b before releasing the NVM semaphore.

Sector Erase Flow:

1. Poll the FLSWCTL.DONE bit until it is set.
2. Set FLSWCTL.CMD field to 0010b and set the FLSWCTL.ADDR field to any address that belongs to the Flash 4 KB sector to be erased.
3. Wait until the FLSWCTL.DONE bit is read as 1b and the FLSWCTL.FLBUSY bit is read as 0b before releasing the NVM semaphore.

Note: Software may erase a sector (using the FLSW* registers) only if the Flash is set to non-secured mode. If the software attempts to erase the whole Flash while in secured mode (nominal operation), the device sets the DONE bit without executing the command. Setting the DONE bit, an interrupt is issued to firmware.

Block Erase Flow:

See "Flash Wear-out Protection" section.

3.3.5.2 Software or Firmware Flow to the Bit-banging Interface

This section is relevant to software only while in the non-secure mode.

To directly access the Flash, software/firmware should follow these steps:

1. Write a 1b to the *Flash Request* bit (FLA.FL_REQ).
2. Read the *Flash Grant* bit (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 FLA register. The exact protocol used depends on the Flash placed on the board and can be found in the appropriate datasheet.
4. Write a 0b to the *Flash Request* bit (FLA.FL_REQ).
5. Following a write or erase instruction, software/firmware 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 Flash datasheet for the OpCode to be used for reading the Status register.

Notes: The bit-banging interface is not expected to be used during normal operation. Software/firmware should instead use the EEPROM-mode when accessing the base sector and the Flash-mode for other sectors.

If software/firmware must use the bit-banging interface in normal operation, it should adhere to the following rules:

- Gain access first to the Flash using the firmware/software semaphore mechanism.
- Minimize the FLA.FL_REQ setting for a single byte/word/Dword access or other method that guarantee fast enough release of the FLA.FL_REQ.

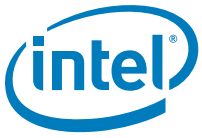
When hardware Flash bit-bang access is aborted due to deadlock avoidance, the *FLA.FLA_ABORT* bit is set. To clear the block condition and enable further access to the Flash, software should write 1b to the *FLA.FLA_CLR_ERR* bit.

3.3.5.3 Software Word Access Flow to the EEPROM-Mode Interface

Software must take semaphore ownership before executing these flows.

3.3.5.3.1 Read Interface

Software initiates a read cycle to the NVM via the EEPROM-mode as follows:



1. Software writes the address to be read in the EERD register
2. Software polls the EERD.DONE bit until it is asserted.
3. Software reads the EERD.DATA register field.

Hardware executes the following steps:

1. Eventually clears the *CMDV* bit if the command cannot be currently executed, and goes to step 4.
2. Reads the data from the shadow RAM.
3. Puts the data in *DATA* field of the EERD register.
4. Sets the *DONE* bit in the EERD register.

Note: Any word read this way is not loaded into Foxville's internal registers. This happens only at a hardware auto-load event.

3.3.5.3.2 Write Interface

Software initiates a write cycle to the NVM via the EEPROM-mode as follows:

1. Poll the *DONE* bit in the EEWR register until it is set.
2. Write the data word and its address in the EEWR register.

As a response, hardware executes the following steps:

1. Eventually clears the *CMDV* bit if the command cannot be currently executed, and goes to step 3.
2. Foxville writes the data to the shadow RAM.
3. Foxville sets the *DONE* bit in the EEWR register.

Notes: The VPD area of the NVM can be accessed only via the PCIe VPD capability structure.

EEPROM-mode writes are performed into the internal shadow RAM. Software can instruct copying of the internal shadow RAM content into the base sector of the Flash device by setting the EEC.FLUPD bit.

3.3.5.4 Flash Program Flow via the Memory Mapped Interface

Software must take semaphore ownership before executing the flow. Software initiates a write cycle via the Flash BAR as follows:

1. Write the data byte to the Flash through the Flash BAR. Use the Byte Enable (BE) pins if less than four bytes has to be written.
2. Poll the FL_BAR_BUSY flag in the FLA register until cleared.
3. Repeat the steps 1 and 2 if multiple bytes should be programmed.

As a response, hardware executes the following steps for each write access:

1. Set the FL_BAR_BUSY bit in the 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_BAR_BUSY bit in the FLA register.

Note: Software must erase the sector prior to programming it. In secured mode, the software may write to the Flash only within the SW Free block and the Free provisioning block.

3.3.5.5 Software or Firmware Flash Program Flow via the Flash-Mode Interface

Software must take semaphore ownership before executing the flow.

1. Poll the FLSWCTL.DONE bit until it is set. This step is only needed if the flow is executed following a reset event.
2. Write the number of bytes to be written into FLSWCNT.CNT field. The write must not cross a page (256 byte) boundary.
3. Set the *ADDR* field with the byte resolution address in the FLSWCTL register and set the *CMD* field to 0001b.
4. Write the data to the FLSWDATA register.
5. Hardware starts accessing the Flash and begins writing data bits from the FLSWDATA register. If the write is not allowed, the *CMDV* bit is cleared instead.
6. Once hardware completes writing the data to the Flash, the FLSWCTL.DONE register bit is set.
7. Hardware increments FLSWCTL.ADDR field by four (Dword granularity) if byte count left is greater or equal to 4.
8. Software polls the FLSWCTL.DONE bit until it is set.
9. Steps 4 to 8 are repeated several times until the number of bytes programmed in FLSWCNT.CNT field has been written.
10. FLSWCTL.GLDONE bit is set by hardware when the last byte programmed has been written. But software can stop the transaction in the middle as long as it got the *DONE* bit read as 1b. In any case, the *FLBUSY* bit must be read as 0b before releasing the NVM semaphore.

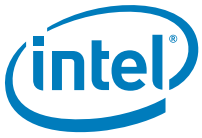
Notes: Firmware uses the FLMNG* registers set instead. Whenever the DONE bit is set by hardware, an interrupt is issued to firmware.
In secured mode, the software may write to the Flash only within the SW Free block and the Free provisioning block.

Other write opcodes can be issued instead of 0001b in CMD field, e.g. 0110b for writing the Write Enable register to the flash device. FLSWCTL.ADDR field is meaningless for such accesses. CMDV bit is cleared by hardware whenever the command is not valid. Refer to the bit description in Section 8.4.9 for details.

3.3.5.6 Software or Firmware Flash Read Flow via the Flash-Mode Interface

Foxville provides an engine for reading the Flash in a burst mode:

1. Poll the FLSWCTL.DONE bit until it is set. This step is only needed if the flow is executed following a reset event.
2. Set the FLSWCNT.CNT field with the number of bytes to be read from Flash in a burst mode.
3. Set the FLSWCTL.ADDR field with the byte address of the first Dword to be read and set the *CMD* field to 0000b. The FLSWCTL.GLDONE bit is cleared by hardware to indicate a burst read has started.
4. Hardware starts accessing the Flash and clears the FLSWCTL.DONE bit until it writes the read Dword into the FLSWDATA register.
5. Software polls the FLSWCTL.DONE bit until it is set.
6. Software reads the Dword from FLSWDATA register, which is used by hardware to trigger a clear of the FLSWCTL.DONE bit again.
7. Hardware increments FLSWCTL.ADDR field by four (Dword granularity) if byte count left is greater or equal to 4.
8. Steps 5 to 7 are repeated until the number of bytes programmed in FLSWCNT.CNT has been read.



9. Hardware sets the FLSWCTL.GLDONE bit to indicate that all the Flash transactions related to the command issued at step 3 were completed. However, software can stop the transaction in the middle as long as it got the *DONE* bit set.

Notes: Firmware uses the FLMNG* registers set instead. Whenever the DONE bit is set by hardware, an interrupt is issued to firmware.

Other read opcodes can be issued instead of 0000b in CMD field, e.g. 0101b for reading the Write Status register from the flash device. FLSWCTL.ADDR field is meaningless for such accesses.

3.3.6 NVM Validity Field

The only way Foxville can tell if a Flash is present and programmed is by trying to read the NVM *Validity* field at NVM word addresses 0x012 and 0x812. If one of the *Validity* fields is read as 01b, a programmed lash is assumed to be present.

3.3.7 Flash Deadlock Avoidance

The Flash is a shared resource between the following clients:

1. Hardware MAC auto-read.
2. Hardware PHY auto-load
3. LAN software accesses.
4. Manageability/firmware accesses.
5. Software tools.

All clients can access the Flash using parallel access, on which hardware implements the actual access to the Flash. Hardware schedules these accesses, avoiding starvation of any client.

However, the software and firmware clients can access the Flash using bit-banging. In this case, there is a request/grant mechanism that locks the Flash to the exclusive use of one client. If one client is stuck without releasing the lock, the other clients can no longer access the Flash. To avoid this deadlock, Foxville implements a timeout mechanism, which releases the grant from a client that holds the Flash bit-bang interface (FLA.FL_SCK bit) for more than 8 seconds. If any client fails to release the Flash interface, hardware clears its grant enabling the other clients to use the interface.

Note: The bit-banging interface does not guarantee fairness between the clients, therefore it should be avoided in normal operation as much as possible. When write accesses to the Flash are required the software or manageability should access the Flash one word at a time releasing the interface after each word. Software and firmware should avoid holding the Flash bit-bang interface for more than 500 ms.

The deadlock timeout mechanism is enabled by the *Deadlock Timeout Enable* bit in the Control Word 1 in the Flash.

3.3.8 VPD Support

The Flash 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 NVM contains a pointer to the VPD area in the Flash. It is recommended to map the VPD area into the RO area of the shadow RAM. Word 0x0A contains the NVM VPD_EN bit, which controls whether or not the VPD capability appears in the configuration space following the next PCI reset event. The VPD_EN bit must be set to 1b only once a valid VPD structure is programmed in the NVM.

The maximum area size is 1024 bytes but 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 3.23. Small Resource Structure

Offset	0	1 — n
Content	Tag = 0xxx,xyyyb (Type = Small(0), Item Name = xxxx, length = yy bytes)	Data

Table 3.24. Large Resource Structure

Offset	0	1 — 2	3 — n
Content	Tag = 1xxx,xxxxb (Type = Large(1), Item Name = xxxxxxxx)	Length	Data

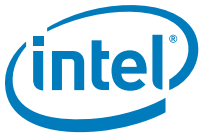
Foxville firmware parses the VPD structure during the auto-load process following PCIe reset in order to detect the read only and read/write area boundaries. Foxville assumes the following VPD fields with the limitations listed:

Table 3.25. 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. The tag must be word aligned.
- If Foxville 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 description of Table 3.25, it assumes the area is not programmed:
  - — Any read/write access through the VPD registers set are ignored.
  - — The VPD pointer itself remains RO.
- The VPD RO area and RW area are both optional and can appear in any order. A single area is supported 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 must be accessed through the PCIe configuration space VPD capability structure listed in Table 3.25. Write accesses to a read only area or any accesses outside of the VPD area via this structure are ignored. The VPD area is also accessible for read via the EEPROM-mode access, and for write via the same access mode - only if the VPD area is mapped to a RW area of the shadow RAM (not recommended).
- VPD area must be mapped to the first valid 4 KB sector of the Flash.



Once firmware completes the parsing of the VPD area, it reports any error detected in VPDDIAG register.

VPD software does not check the semaphores before attempting to access the Flash via dedicated VPD registers. Even if the Flash is owned by another entity, VPD software read or write access directed to the VPD area might complete immediately since it is first performed against the shadow RAM. Firmware is responsible for handling the VPD read and write accesses against the host. Hardware notifies firmware each time a VPD access was initiated. However, VPD software write access is written into the Flash device at the firmware's initiative, which might take up to several seconds. Refer to Section 3.3.10.3.

3.3.9 NVM Protection and Security

The NVM contains parameters that are expected to be programmed by the customers (either the platform or NIC manufacturer or by the end user on the field). There are also parameters that are expected to be programmed by Intel® or at least approved by Intel®. these parameters are stored in write protected spaces in the NVM or secured modules that are signed by Intel® using a cryptographic scheme. The following sections describes these modules and parameters.

3.3.9.1 Write Protection of the Shadow RAM Sectors

The shadow RAM sectors are stored in the first 2 x 4KB sectors in the NVM. Only one of these two sectors is active at a time and it is loaded to an internal shadow RAM. Software can modify the content of these sectors ONLY by accessing the shadow RAM which is done using the EEPROM-Mode registers. Some of the parameters in the shadow RAM are considered as "sensitive" and are write protected for the software. These parameters can be still be modified as part of a complete update of a signed image (see additional details in Section 3.3.10.2). Listed below are the write protected fields in the shadow RAM:

1. The first protected area is a segment defined by the "Start of RO Area pointer" and "End of RO Area pointer" words in the NVM (at addresses 0x002D and 0x002C respectively). The modules that are located in this area are listed between these two pointers in Table 3-20.
2. The second protected area is located at the end of the 4 KB shadow RAM. It is defined by the "Start of 2nd protected area" field in the NVM (at address 0x0012).
3. On top of it, all modules pointers and module sizes, the NVM validity flag and some additional parameters are write protected parameters. For the complete list of these unique protected parameters see the Table 3-26 below.

Table 3-26. NVM Protected Words in the Shadow RAM Space

Word Offset	Word Name	Word Offset	Word Name
0x00..0x02	Ethernet Individual MAC Address	0x2C	End of Read-Only (RO) Area
0x03	Compatibility Bytes	0x2D	Start of RO Area
0x08	PBA Number 0	0x2F	VPD Pointer
0x09	PBA Number 1	0x34	PCIe PHY Configuration 0 Low
0x0D	Device ID	0x35	PCIe PHY Configuration 0 High
0x0E	Vendor ID	0x37	Alternate MAC Address Location
0x0A	Initialization Control Word 1	0x38	PCIe PHY Configuration 1 Low
0x10	Secure area start address	0x39	PCIe PHY Configuration 1 High
0x11	Flash Device Size	0x3A	PCIe PHY Configuration 2 Low / Reset to PCIe PHY Delay (x40us)
0x12	EEPROM Sizing and Protected Fields	0x3C	PXE VLAN pointer

Table 3-26. NVM Protected Words in the Shadow RAM Space

Word Offset	Word Name	Word Offset	Word Name
0x17	SW Reset CSR Auto Configuration Pointer	0x3D	iSCSI Boot Configuration Pointer
0x20	Software Defined Pins Control	0x40	Free Provisioning Area Pointer
0x22	LAN Power Consumption	0x41	Free Provisioning Area Size
0x23	PCIe Reset CSR Auto Configuration Pointer	0x44	PCIe L1 Substates Capability Low
0x24	Initialization Control 3	0x45	PCIe L1 Substates Capability High
0x25	mDNS Records Area Offset	0x46	PCIe L1 Substates Control 1st Low
0x26	mDNS Records Area Size	0x47	PCIe L1 Substates Control 1st High
0x27	CSR Auto Configuration Power-Up Pointer	0x48	PCIe L1 Substates Control 2nd
0x28	PCIe Control 2	0x4A	EXP. ROM Boot Code Section Pointer
0x2A	CDQM Memory Base Low	0x50	RO Commands Version
0x2B	CDQM Memory Base High	0x51	Firmware Module Configuration Pointer

3.3.9.2 Write Protection of the VPD Structure

The VPD structure is stored in the Shadow RAM sectors. Accessing this structure by the EEPROM-Mode registers, this structure is write protected. Software can program this structure using the standard interface in the VPD capability in the PCIe configuration space.

3.3.9.3 Secured NVM Structures

The secured Block containing the CSS header, FW content, PHY Firmware and the OROM module. There are 2 possible locations in the NVM for these modules in "block 2" or "block 3" in the NVM as illustrated in Figure 3-5. The active block is write protected while the inactive block can be updated by the software. Before this updated image is activated, Foxville checks its signature to verify that it is indeed authentic. More details on the secured structures and image update method are described in Section 3.3.10.1.

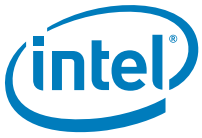
3.3.10 NVM Update Flows

3.3.10.1 Flow for Updating the Firmware/OROM/PHY Secured Modules

In order to protect the Flash update procedure from power-down events, a double image policy is required each time the Secured image is updated. The software/Firmware should proceed as follows:

Proceeding steps to the actual programming:

1. **A new Secured Module is created by Intel PAE**. The last sector in the FW module contains commands to FW for updating the RO items, it is referred as the RO Updates section. The sector before the last sector contains the list of the supported Flash devices that require special opcodes, it is referred as the Flash Devices Table. Format of the RO Updates is described in Section 6.1.24, and format of the Flash Device Table (see Section 3.3.11.2). The first word of these two sections contains the Version number which should be incremented whenever the section is modified.
2. **The new Secured Module is sent to Intel CSS** for adding to it the new authentication signature. The authentication covers the whole Secured Module.
3. **The new Secured Module returned from CSS is issued to the customer**.



Read the pointer to the Secured Block (word 0x10), which is the current location of the module, and read the pointer to the free provisioning area (word 0x40).

1. Software initiate a request from the firmware to erase the free provisioning block. Refer to Section 3.3.5.1.
2. The software takes ownership over the NVM via the semaphore bits. Refer to Section 4.8.
3. Software writes the new secured Block 3 modules in the free provisioning area by the Flash-mode access (memory mapped or FLSW* register set).
   a. It is recommended to write at this step no more than four 4 KB sectors at once in a burst, releasing semaphore ownership for 10 ms in between.
4. Release the NVM semaphore and set the FLFWUPDATE.Update bit to 1b.
   a. Software must avoid taking the NVM semaphore again until the firmware resets and reloads from the new image. Any new attempt to write the NVM until then is not performed by the device.
5. Setting the FLFWUPDATE.Update bit, an interrupt is generated to the firmware.
6. Firmware checks the following. If any of the checks below fail, go to the "Done" step.
   a. Check that the Foxville Blank NVM Device ID field of the new secured module header matches the blank Device ID, whatever SKU it may be.
   b. Check the lad_srev fields of the new secured module header is greater or equal to the lad_srev field of the current secured module header.
   c. Check that the size field in the CSS header matches the expected length.
   d. Check that the RSA Modulus and RSA Exponent fields in the new FW image are identical to the fields in the old FW image.
   e. Perform the NVM Authentication Procedure described in Section 3.3.11, using the key materials stored in the old FW module.
7. Firmware swaps between Block 3 pointer (word 0x10) and Block 4 pointer (word 0x40) in the shadow RAM: the secured block and free Provisioning block.
8. Firmware dumps the shadow RAM into the Flash. by the flow described in Section 3.3.2. Note that in case the new FW image has an updated Read Only section (the first word of the RO Updates section is different than 0xFFFF) then execute the "Updating a RO NVM" flow described in Section 3.3.10.2. If on top of it, the updated RO section has *new* RO type(s), the FW image should be updated twice as follow:
   1) Load a new signed FW image built with the *old* RO types.
   2) Load the new signed FW image with the *new* RO types.
9. Firmware sets the FLFWUPDATE.AUTH-DONE bit to notify software that the firmware update procedure has completed.
10. Firmware resets itself and reload from the new firmware image.
    a. Note that the new RO settings handled at step 9 are loaded into the device before the new firmware code starts running.
11. Software polls the FLFWUPDATE.AUTH-DONE bit until it is read as 1b.
    a. If FLFWUPDATE.AUT_FAIL bit is read as 1b, it means that the update process failed because of one of the security checks has failed or because of some defect in the Flash write. In such case software exits the flow and might decide to rerun it from the beginning.

Note: When firmware executes that flow, the 10 ms waiting time between two consecutive NVM semaphore ownership taking cycles is configurable by Semaphore Backoff Interval field in the Common Firmware Parameters 2 NVM word.

Note: The device may not reply to requests from the MC during the firmware update process, which can last up to 3 seconds.

3.3.10.2 Flow for Updating a RO NVM Item

The following flow is performed to modify 'on the field' any RO item specified in Table 6.1 and/or to modify the contents of the Flash Devices Table included in the trailer of the Firmware Secured Module.This flow is executed as part of an update of the secured NVM image detailed in Section 3.3.10.1.

1. **Old FW takes NVM ownership**.
2. **Old FW writes the changes to the RO items into shadow RAM** - it parses the new RO Updates read from the last FW image sector, and writes the required changes into the shadow RAM. Section 6.9.2 for the RO Updates formats. In case the entire shadow RAM contents has to be rewritten, it is done except for:
   a. The content of the Secured Module Pointer (word 0x10) which is not overridden.
   b. The Checksum (word 0x3F) which is recomputed and updated.
3. **Old FW updates the RO Updates Version** - it copies the RO Updates Version field read from the new RO Update section into the shadow RAM word 0x50. The old RO Update Version is stored in the FW RAM before.
4. **Old FW erases next 4KB bank** - it checks that the next bank contents is all 1s. If it is not, then it erases the contents of the next basic bank sector.
5. **Old FW copies shadow RAM into next 4KB bank** - it copies the shadow RAM into the next basic bank sectors – excepted to the NVM Validity field left all 1s. RO items are also copied from the shadow RAM contents. Checksum (word 0x3F) shall be updated by EMP in both shadow RAM and next basic bank.
6. **Old FW 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 at previous step.
   a. **If it is not identical, FW restores the old RO Update Version or toggles it to 0xFFFF in case it was 0x0000, and exits this flow** - it restores the old RO Updates Version stored at step 3. into the shadow RAM, and resumes its update flow of Section 3.3.10.1.
   b. **If the check was ok, FW validates the new bank and invalidates the old bank** -
   — NVM Validity field of the new bank is set to 01b. FW checks the NVM Validity field is read as written in the flash; if it is not, then go to the previous sub-step.
   — FW toggles the state of the SEC1VAL bit in the EEC register to indicate that the non-valid bank became the valid one and vice versa.
   — The current (old) bank is invalidated by setting its NVM Validity field to 00b.
7. **Old FW releases NVM ownership**.
8. **Old FW resumes its update flow** of Section 3.3.10.1.

Note: Depending on the modified RO items, a system reset is generally required for loading the modifications into the device.

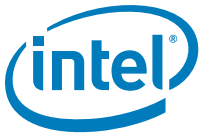
In case the RO Updates section contained a full Shadow RAM contents update, Software Tools are responsible to restore the customized RW items in the shadow RAM after completion of this flow.

3.3.10.3 VPD Write Flows

3.3.10.3.1 First VPD Area Programming

The VPD capability is exposed on the PCIe interface only if the VPD_EN bit in NVM word 0x0A is set to 1b, regardless of any other sanity check that is performed on the VPD area contents.

The VPD contents and pointer can be written on a blank Flash without any limitation, similar to any other NVM module when in the blank Flash programming mode. This is the recommended way to map the VPD area into the RO protected area of the shadow RAM, which is highly advised.



It is also useful to provide also a means for performing the initial VPD area programming on a non-blank flash (i.e. NVM Validity field in NVM word 0x12 read as 01b). In such a case, the VPD pointer (NVM word 0x2F) shall point to the provisioned (empty) VPD area, set in the first NVM programming of the blank flash. If the VPD area is already programmed, the flows described in Section 3.3.10.3.2 or in Section 3.3.10.2 shall be used to modify or reprogram it, respectively.

The first VPD area programming flow on a non-blank flash is basically the same flow that is used for writing RW words in the NVM Header (see Section 3.3.10.4). The following points shall be taken into consideration:

1. Take ownership over the NVM via semaphore bits. Refer to Section 4.8.
2. Write the contents of the VPD module via EEPROM-Mode access. The VPD area shall be written within the RW shadow RAM limits, over an unused RW shadow RAM segment which does not overlap over other existing modules. It is assumed that the VPD pointer at word 0x2F already points to the start of such a provisioned VPD area.
3. Write the VPD_EN bit to 1b at word 0x0A via EEPROM-Mode access.
4. Release the NVM semaphore.
5. Set the *FLUPD* bit in EEC register to ask the device firmware to load the internal shadow RAM into the Flash.
6. Firmware dumps the shadow RAM into the flash via the flow described in Section 3.3.2.
7. Poll the *FLUDONE* bit in EEC register until it is set by the device firmware.
8. The VPD capability will be exposed after the next PCIe reset or POR event.

3.3.10.3.2 VPD Area Update Flow

1. The host performs a VPD write - it sets write offset/data into VPD register set of the configuration space, setting the VPD Flag (bit 15 in VPD Address Register - 0x0E2).
2. Hardware notifies FW - it issues a internal 'VPD access' interrupt to FW to notify it of the VPD access.
3. Firmware checks the VPD write is allowed - it checks that the write offset points to the VPD-RW area and not to the RO area of the shadow RAM and nor to the VPD-RO area.
   a. If it is not, firmware clears the VPD flag in the configuration space to notify the VPD software that the transaction completed, and exits the flow.
4. Firmware takes NVM semaphore ownership.
5. Firmware re-starts the 10 ms VPD timer and writes the change into shadow RAM.
6. Firmware completes the VPD access to software - firmware clears the VPD flag in the configuration space to notify the VPD software that the access completed.
7. Firmware releases NVM semaphore ownership.
8. When the VPD timer expires, firmware dumps the shadow RAM into the Flash according to the flow described in Section 3.3.2.

   If VPD write access is attempted by the host when the device has just started a Flash erase operation, or if NVM ownership is held by software for a long time, then the VPD write request might time out as the firmware code responsible to handle the request would not be readable from the NVM. As a result, Intel recommends that a software application that modifies the VPD area perform back-to-back VPD write accesses within no longer than 10 ms between two consecutive writes.

3.3.10.4 Flow for Updating One of the RW Legacy EEPROM Modules

When updating one or several fields from a legacy EEPROM module there is a risk that a hardware auto-load event occurs in the middle of the operation (due to a sudden PCIe reset for instance), leading to the auto-load of an invalid or inconsistent content from the internal shadow RAM into the device

registers or memory. Therefore unless the field(s) can be updated by a single EEPROM-mode access, the updating software/firmware must repeatedly use the following procedure for each legacy EEPROM module to be updated:

1. Take ownership over the NVM via semaphore bits. Refer to Section 4.8.
   a. In case the NVM update originates from a BMC command, firmware will attempt to take NVM ownership once every 5 ms up to 40 ms, until it get the ownership. If it does not succeed to get NVM ownership until 40 ms, firmware will complete the BMC command with a 'package not ready' status.
2. Invalidate the pointer to the module to be modified by setting it to 0xFFFF via EEPROM-mode access. This way, if a hardware auto-load of the module is attempted, the associated register defaults are loaded instead. Do not invalidate pointers to firmware modules.
3. Modify the contents of the module via EEPROM-mode access.
4. Restore the pointers to the modified module(s) via EEPROM-mode access.
5. Compute and update the software checksum (word 0x3F) if the contents covered by the software checksum was modified.
6. Release the NVM semaphore.
7. Set the *FLUPD* bit in EEC register to ask the device firmware to load the internal shadow RAM into the Flash.
8. Firmware dumps the shadow RAM into the Flash via the flow described in Section 3.3.2.
9. Poll the *FLUDONE* bit in EEC register until it is set by the device firmware.

Note: Depending on the modified RO items, a system reset is generally required for loading the modifications into the device.

3.3.11 NVM Security

The NVM update integrity feature ensures that only Intel approved firmware code (or another protected NVM module) is able to be updated on Foxville devices after manufacturing. This procedure is performed by a flash-based firmware code (with no hardware acceleration) whenever attempting to update one of the protected modules. Refer to NVM update flows in Section 3.3.10.

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 (long by 256-bits), which 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 (the public key) and RSA Exponent parameters to be used for decrypting the digital signature.

To verify the authenticity of the digital signature, firmware must first verify that the RSA Modulus and RSA Exponent fields in the new secured image loaded are identical to those in the old FW image. If the RSA Modulus and Exponent fields are the same, firmware decrypts the digital signature using the 2048-bit RSA Modulus and Exponent fields stored in the manifest of the old secured image to extract the expected SHA256 Hash of content (stored hash). Firmware 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 update is applied.

The RSA signature keys used to generate the signature is part of the Intel Code Signing System (CSS) that is also used to sign other sensitive Intel firmware such as Manageability Engine (ME) firmware.

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 firmware switches to the new image.

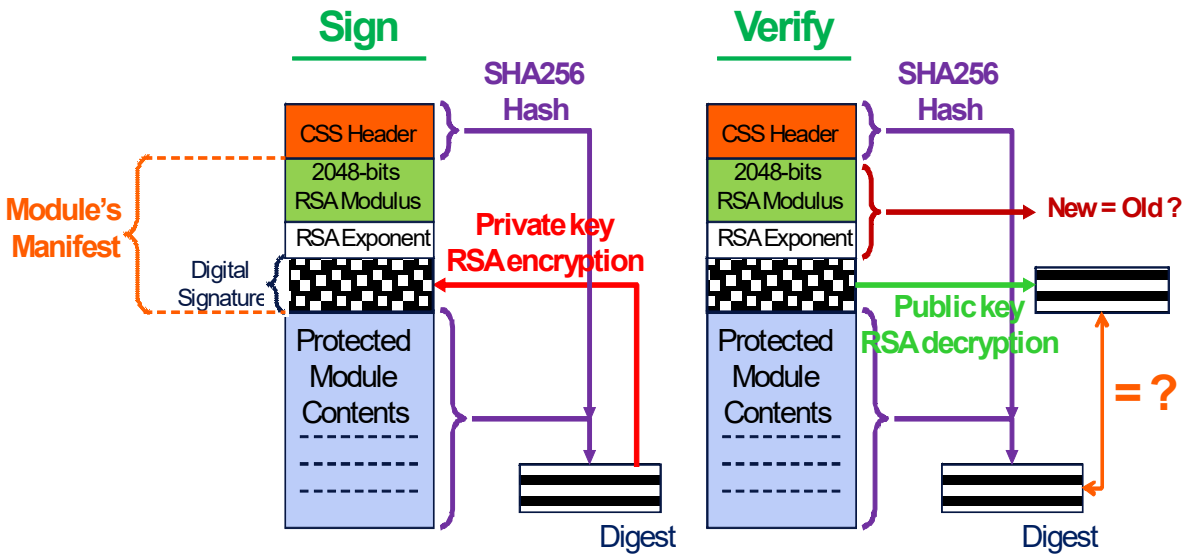


Figure 3-7. Sign and Verify Procedures for Authenticated NVM Modules

3.3.11.1 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 Foxville.

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 Foxville blank NVM Device ID word of the NVM header described in Section 6.1.25, and ends with the last word of the Secured Module - regardless to the size of the firmware code and to the presence and size of a Flash Devices Table and RO Updates sections at the last two sector of the Firmware Secured Module area.

3.3.11.2 Intel Key Generation and the 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, and a method to generate new signed images using the existing private keys.

The Intel Code Signing System (iCSS) is utilized by various product teams within Intel. The high-level web site portal into this system is located at http://moss.amr.ith.intel.com/sites/DTS-CCT-W/DATA/ltcss/default.aspx. Utilization of this service is allowed to the point that LAD does not make feature modification requests of the system - e.g. LAD's products must conform to the existing processes and formats required of the system, as well as utilize the algorithms supported by the system.

CSS mandates a number of groups, roles, and group membership be defined when a project is created. This task would normally fall to the LAD program management to facilitate, as any group which needs to create and sign images would need access and appropriate approvals. The system also requires acquisition and distribution of special-purpose USB "Rainbow" token keys (~ $50 / pc in volume) as a physical verification of the user above and beyond passwords. More information is available from the CSS User's Guide.

Using the system would require some degree of training to any user who needs to generate signed images - including self-signed images developers and validators would produce.

Note: CSS does not make the productized private keys available to LAD - the private keys are only used internal to the CSS system when used to generate new signed NVM images.

The Intel Code Signing System follows the PKCSv1.5 format with 2048-bit RSA keys and SHA256 hash. The CSS algorithm requires a standard manifest header to appear at the beginning of the signed module. The manifest header (represented in "C" syntax) is as follows:

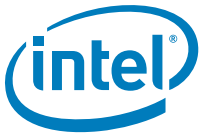
```
typedef struct _CssHeader {
    uint32 moduleType;          // Reserved CSS field
    uint32 headerLen;           // Reserved CSS field
    uint32 headerVersion;       // Reserved CSS field
    uint32 moduleID;            // Reserved CSS field
    uint32 moduleVendor;        // Reserved CSS field
    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;            // LAD-specific security revision field
    uint32 reserved;         // Reserved for future use
    uint32 lad_fw_entry_offset;// LAD-specific offset from start of CSS header of the firmware
code in WORDS
    uint32 lad_fl_dev_offset; // LAD-specific offset from start of CSS header of the flash devices
parameters table in WORDS

uint32 lad_image_unique_id; // LAD-specific unique identifier for the specific NVM image version.
Taken from NVM words 0x42, 0x43 (a.k.a. EETRACK ID)
    int32 reserved[17];        // Mandatory field, but free for use
} CssHeader;
```

The CSS header must be placed at the beginning of the integrity-checked data. A software tool is required to pre-process raw NVM images to prepend the manifest for submission to the CSS tool chain. All fields marked *Reserved CSS field* must be zero on submission to the CSS tool chain.

Foxville CSS header includes a *Security Revision* (lad_srev) field. The *Security Revision* 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 released with an incremented security revision, Foxville does not allow NVM image roll-back to previous versions (to re-expose known vulnerabilities).

Note: Not all NVM updates need to have an incremented Security Revision - rollback of updates to non-security related parameters - where the *Security Revision* field is equal to the existing image - are allowed.

From the note in Section 3.3.11.1, it comes out that the size is always set to 0x1C000 in Foxville.



3.3.11.3 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 NVM. The device is incapable of generating a signed image by itself because the private key is not available to the device to generate the digital signature in NVM.

Only the following NVM module requires authentication in Foxville. The module includes its own digital signature: secured image and its associated sub-modules.

3.3.11.4 Software Requirements

A software tool MUST prepare NVM images for the CSS signing step, prepending the CSS manifest, 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 the merge was successful (such as a software computation of the digital signature passes).

Software Validation (SV) tools (e.g. lanconf) MUST implement an NVM Update Image integrity check option in software prior to applying NVM updates to hardware. SV tools MAY 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 FLA.LOCKED bit state as part of manufacturing qualification. If after the NVM is programmed, and the device powers up with the 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 FLA unlocked, unauthorized in-the-field updates can bypass designed firmware integrity checks.

Host drivers MAY implement an interface allowing a network administrator to perform an internal verification check of the signed NVM image. On Windows drivers, this would take the form of an OID which reports SUCCESS or INVALID_PARAMETER. On Linux, an ethtool command extension is advised to allow command line interrogation of the NVM content using the hash value build into the HW as well as the saved CSS manifest in the NVM image.

3.3.11.5 Manufacturing Requirements

3.3.11.5.1 Debug and Production Keys

To simplify the debug and validation process prior to PRQ, pre-production devices will be programmed during manufacture with a well-known 'debug' hash (and corresponding 'debug' public key). This debug key is generally available to any internal developers who need to self-sign their own NVM images (and do not require external approvals to create authenticated NVM images).

As a step during promotion of the final product to ship-readiness, the final products are programmed during manufacture with a 'production' hash (and corresponding production public key). This production key is only known to the Code Signing System, and only the Code Signing System is capable of producing the productized signed NVM images. Developers are not able to self-sign NVM images with these production parts.

3.3.11.5.2 OEM Customization

While not a requirement, devices intended for specific OEMs (who intend to activate OEM-specific features in the device) SHOULD have an OEM-specific public key/hash value with an corresponding NVM image signed for specific OEMs devices. This prevents the possibility that invalid or inappropriate NVM updates are applied to OEM devices (e.g. an image specifically for a NIC cannot be loaded on an OEM-LOM design which disables OEM features as a side-effect). It also provides a degree of inventory traceability in the event gray market NICs are produced using Foxville silicon.

3.3.11.5.3 End-of-Line Verification

OEM's MUST execute CELO (or equivalent manufacturing diagnostics tool) to verify the FLA.LOCKED bit state (as described above).

3.3.11.5.4 Post-Manufacturing Physical Modification Countermeasures

For add-in NICs and related form factors, application of a conformal epoxy encapsulation coating around the edges of NVM component (the flash part) epoxy during manufacturing prevents ready removal of components.

In addition, epoxy is used to coat any traces or vias which could enable direct-probing of the device. The testing requirement for these coatings is they must not be easy to remove with a standard hobby knife or other hand-held tools.

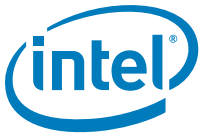
Protection by epoxy encapsulation is just a recommendation, not a security requirement for Foxville.

3.3.12 NVM Init Flows

3.3.12.1 Hardware Auto-load from NVM

This is the first NVM access procedure performed after the device powers up (LAN_POWER_GOOD event). It is initiated by hardware even before the firmware is up.

1. **Look for a flash device** - Read the Status register of the flash device making use of the read status opcode (0x05).
   a. If 0xFF is read, it means that no flash device is connected. Clear the FLASH_DETECTED and FLASH_IN_USE bits in EEC register. :
   b. Otherwise, load the value read in the SHADOWINF.Power_Up_Status register field. Set the FLASH_DETECTED in EEC register.
2. **Look for a valid NVM image** - Read the NVM Validity field of each one of the basic sectors by making use of the read opcode (0x03).
   a. If at least one NVM Validity field is read as 01b, then store the lowest sector index with the NVM Validity field read as 01b into the SEC1VAL bit in the EEC register, and set the FLASH_IN_USE and EE_PRES bits in EEC.
3. If none of the NVM Validity fields is read as 01b, then enter the blank flash programming mode and skip the following steps, clearing the .**Read and store the Flash-ID** - Read the Flash-ID (a.k.a. JEDEC-ID of the flash part) by issuing the Read JEDEC-ID instruction (0x9F) to the flash part. Load the 3-bytes long Flash-ID read into SHADOWINF.JEDEC_ID register field.
4. **Copy the contents of the valid basic sector into the shadow RAM** - Use the flash read opcode (0x03) to read the valid basic sector from flash part.
5. **Perform hardware NVM auto-load** - Load the NVM settings relevant to POR from the shadow RAM into the device registers and memories. Once PCIe/PHY analog settings are loaded, remove the flash device write protection by issuing the write-protection removal OpCode to the Flash.
6. **Exit Firmware from reset** for **NVM Integrity Check** - Check the integrity of the secured NVM structure. It includes the FW module + PHY image + optional ROM image (if exist). The secured NVM structures is described in Section 3.3.9.3.
   a. If integrity check is OK then proceed to the next steps.
   b. Else (the integrity check fails), then disable the embedded Firmware and enable the NVM for re-programming. As a result, the FLA.LOCKED bit is cleared and the EEC.FLASH_IN_USE bit remains high.
7. **Execute Nominal Firmware Code** - Once the NVM integrity check pass OK, the embedded management controller start executing its code from the NVM. As part of the Firmware initialization code, the Firmware release the non-protected fields in the NVM for programming by the software



8. **Auto-Load the PHY Firmware** - Load the PHY firmware from the NVM to the PHY cluster.
9. **Set the EEC. AUTO_RD bit** to notify software and firmware that all the required hardware auto-load processes have been completed. The bit will be set once completing the last auto-load process of the reset chain.
10. **Software Driver Init Flow** - Software is expected to read the :
    a.
    b. Iisre-program the whole NVM. Once it is completed, the software is expected to initiate a Firmware reset (in the HICR register) and then initiate a system boot or initiate a power cycle to the whole system.
    c. The Iis

After a PCIe reset, Foxville reads the global device parameters from the shadow RAM (assuming it has a valid NVM Validity field contents) including all the parameters impacting the content of the PCIe configuration space. After a software reset to the port (*CTRL.DEV_RST* set to 1b), a partial load is done of the parameters relevant to the port where the software reset occurred.

Table 3-21 lists the words in the shadow RAM that are used to initialize device registers at reset events.

3.3.13 FLASH Wear-out Protection

Foxville protects against possible malicious software attempting to wear-out the Flash.

The Flash wear-out protection is implemented by the Firmware. The protection mechanism is efficient and yet very basic:

A device with an empty (or invalid image) is woken in a non-protected mode. In this mode of operation, software is cable to do everything without any protection. Please note that operating with an empty Flash is expected only at production floor or when replacing a Flash device on the field. In both cases the system is expected to be in a relative controlled environment.

During nominal operation, the Flash is valid containing secured content including the embedded firmware. At this mode of operation, the software is permitted to program (write) only to non-secured spaces: the free provisioning block and the software free block (see Figure 3-5 for “block” description). And it has limitations on the erase options:

The software cannot initiate a Flash erase. Initiating a Flash erase dose not impact the NVM content.

The software can initiate a Flash sector erase with the following restrictions:

- The software can initiate a Flash sector erase only to non-protected blocks: Software Free Space and mDNS space as well as the Free Provisioning area. Initiating a sector erase to protected blocks dose not impact the NVM content.
- The rate of initiating the Flash Sector erases is limited. It guaranty that software could not initiate more than 100K sector erases to the same sector during the guaranteed device life time. it is guaranteed in the following manner:
  — the software gets 7 credits per block erase. Note that a block might be consist of multiple sectors.
  — Whenever there are credits (to a specific block), the software can initiate the sector erase for sectors of a block. The software should erase the sectors of the block sequentially, from the lowest sector up to the last sector of the block. All this process consume a single credit of the block.
  — Every 2 hours a new credit is gained up to a limit of 7 credits per block.
  — It is the software responsibility to track the credits. And software should NOT initiate a sector erase if there are no credits for the block.

— If in doubt, the software can query for the credits by the Get Wear-Out Credits host slave command.

3.4 Configurable I/O Pins

3.4.1 General-Purpose I/O (Software-Definable Pins)

Foxville has four software-defined pins (SDP pins) that can be used for miscellaneous hardware or software-controllable purposes. These pins can each be individually configurable to act as either input or output pins. The default direction of each of the four pins is configurable via the NVM as well as the default value of any pins configured as outputs. To avoid signal contention, all four pins are set as input pins until after the NVM configuration has been loaded.

In addition to all four pins being individually configurable as inputs or outputs, they can be configured for use as General-Purpose Interrupt (GPI) inputs. To act as GPI pins, the desired pins must be configured as inputs. A separate GPI interrupt-detection enable is then used to enable rising-edge detection of the input pin (rising-edge detection occurs by comparing values sampled at the internal clock rate as opposed to an edge-detection circuit). When detected, a corresponding GPI interrupt is indicated in the Interrupt Cause register.

The use, direction, and values of SDP pins are controlled and accessed using fields in the Device Control (CTRL) register and Extended Device Control (CTRL_EXT) register.

The SDPs can be used for special purpose mechanisms such as a watchdog indication (refer to Section 3.4.2), IEEE 1588 support (refer to Section 7.5.1).

3.4.2 Software Watchdog

In some situations it might be useful to give an indication to manageability firmware or to external devices that Foxville hardware or the software device driver is not functional. For example, in a pass-through NIC, Foxville might be bypassed if it is not functional. In order to provide this functionality, a watchdog mechanism is used. This mechanism can be enabled by default, according to NVM configuration.

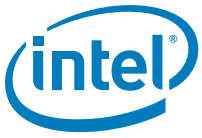
Once the host driver is up and it determines that hardware is functional, it might reset the watchdog timer to indicate that Foxville is functional. The software device driver should then re-arm the timer periodically. If the timer is not re-armed after pre-programmed timeout, an interrupt is sent to firmware and a pre-programmed SDP0 pin is asserted. Additionally the *ICR.Software WD* bit can be set to give an interrupt to the software device driver when the timeout is reached.

The SDP0 pin on which the watchdog timeout is indicated, is defined via the *CTRL.SDP0_WDE bit*. In this mode, the *CTRL.SDP0_IODIR* should be set to output. The *CTRL.SDP0_DATA* bit indicates the polarity of the indication. Setting the *CTRL.SDP0_WDE* bit causes the watchdog timeout indication to be routed to this SDP0 pin.

The register controlling the watchdog timeout feature is the *WDSTP* register. This register enables defining a time-out period and the activation of this mode. Default watchdog timeout activation and timeout period can be set in the NVM.

The timer is re-armed by setting the *WDSWSTS.Dev_functional* bit.

If software needs to trigger the watchdog immediately because it suspects hardware is stuck, it can set the *WDSWSTS.Force_WD* bit. It can also supply firmware the cause for the watchdog, by placing additional information in the *WDSWSTS.Stuck Reason* field.



Note: The watchdog circuitry has no logic to detect if hardware is not functional. If the hardware is not functional, the watchdog might expire due to software not being able to access the hardware, thus indicating there is potential hardware problem.

3.4.2.1 Watchdog Re-arm

After a watchdog indication was received, in order to re-arm the mechanism the following flow should be used:

1. Clear *WD_enable* bit in the WDSTP register.
2. Clear *SDP0_WDE* bit in CTRL register.
3. Set *SDP0_WDE* bit in CTRL register.
4. Set *WD_enable* bit in the WDSTP register.

3.4.3 Configurable LED Outputs

Foxville provides three LEDs / GPIO pins that can be used to indicate different statuses of the link. The default setup of the LEDs is loaded from the NVM word offsets 0x1C and 0x1F. This setup is reflected in the LEDCTL register. The software device driver can change this setup accessing the LEDCTL register. For each of the LEDs, the following parameters can be defined:

- LED Functionality (Mode): Defines which information is reflected by this LED as shown in the table below.
- Polarity (IVRT): Defines the polarity of the LED.
- Blink mode (BLINK): Determines whether or not the LED should blink or be stable. In addition, the blink rate of all LEDs can be defined by the "Global Blink Mode" flag in the LEDCTL register. The possible rates are 200 ms or 83 ms for each phase. There is one rate for all the LEDs.

Table 3-27. Link Mode Encoding

LEDx_MODE (x=0...3)	LED Functionality
0x0	LED_ON - Always high (Asserted)
0x1	LED_OFF - Always low (De-asserted)
0x2	LINK_UP - Asserted when any speed link is established
0x3	FILTER_ACTIVITY - Asserted when link is established and packets are being transmitted or received that passed MAC filtering
0x4	LINK_ACTIVITY - Asserted when link is established and when there is no transmit or receive activity. When BLINK is set, the LED is on if there is link and it blinks for activity (either receive or transmit).
0x5	LINK_10 - Asserted when a 10 Mb/s link is established
0x6	LINK_100 - Asserted when a 100 Mb/s link is established
0x7	LINK_1000 - Asserted when a 1000 Mb/s link is established
0x8	LINK_2500 - Asserted when a 2500 Mb/s link is established
0x9	SDP_MODE - The SDP pin functions as an SDP. SDP0...2 on LED 0...2 pins respectively.
0xA	PAUSED - Asserted when the transmitter is PAUSED by the flow scheme
0xB	ACTIVITY - Asserted when link is established and packets are being transmitted or received
0xC	LINK_10/100 - Asserted when either 10 or 100 Mb/s link is established
0xD	LINK_100/1000 - Asserted when either 100 or 1000 Mb/s link is established and maintained
0xE	LINK_1000/2500 - Asserted when either 1000 or 2500 Mb/s link is established
0xF	LINK_100/2500 - Asserted when either 100 or 2500 Mb/s link is established and maintained
The dynamic LED modes (FILTER_ACTIVITY, LINK/ACTIVITY, ACTIVITY, PAUSED) should be used with LED Blink mode enabled.	

3.5 Internal Voltage Regulators

See the GPY211 PHY documentation

3.6 Network Interfaces

3.6.1 Overview

Foxville MAC and an integrated PHY provide a complete CSMA/CD function supporting IEEE 802.3 (10 Mb/s), 802.3u (100 Mb/s), 802.3z and 802.3ab (1000 Mb/s) and 802.3bz (2.5 Gb/s) implementations. Foxville performs all of the functions required for transmission, reception, and collision handling called out in the standards.

Foxville's copper PHY supports 10/100/1000/2500 BASE-T signaling and is capable of performing intelligent power-management based on both the system power-state and LAN energy-detection (detecting unplugged cables). Power management includes the ability to shut-down to an extremely low (powered-down) state when not needed, as well as the ability to auto-negotiate to lower-speed (and less power-hungry) 10/100 Mb/s operation when the system is in low power-states.

3.6.2 MAC Functionality

3.6.2.1 Internal GMII/MII Interface

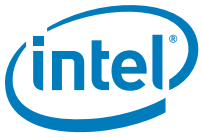
Foxville's MAC and PHY/PCS communicate through an internal GMII/MII interface that can be configured for either 2.5 Gb/s (GMII @ 312.5MHz), 1000 Mb/s operation (GMII @ 125MHz) or 10/100 Mb/s (MII) mode of operation. For proper network operation, both the MAC and PHY must be properly configured (either explicitly via software or via hardware auto-negotiation) to identical speed and duplex settings.

All MAC configuration is performed using Device Control registers mapped into system memory or I/O space; an internal MDIO/MDC interface, accessible via software, is used to configure the internal PHY.

3.6.2.1.1 Other MAC/PHY Control/Status

In addition to the internal GMII/MII communication and MDIO interface between the MAC and the PHY, Foxville implements a handful of additional internal signals between MAC and internal PHY, which provide richer control and features.

- PHY reset: The MAC provides an internal reset to the PHY. This signal combines the PCI_RST_N input from the PCI bus and the PHY *Reset* bit of the Device Control register (CTRL.PHY_RST).
- PHY link status indication: The PHY provides a direct internal indication of link status (LINK) to the MAC indicating whether it has sensed a valid link partner. Unless the PHY has been configured via its MII management registers to assert this indication unconditionally, this signal is a valid indication of whether a link is present. The MAC relies on this internal indication to reflect the *STATUS.LU* status, as well as to initiate actions such as generating interrupts on link status changes, re-initiating link speed sense, etc.
- PHY duplex indication: The PHY provides a direct internal indication to the MAC of its resolved duplex mode (FDX). Normally, auto-negotiation by the PHY enables the PHY to resolve full/duplex communications with the link partner (except when the PHY is forced through MII register settings). The MAC normally uses this signal after a link loss/restore to ensure that the MAC is configured consistently with the re-linked PHY settings. This indication is effectively visible through the MAC register bit *STATUS.FD*, each time MAC speed has not been forced.



- PHY speed indication(s): The PHY provides direct internal indications (SPD_IND) to the MAC of its negotiated speed (10/100/1000/2500 Mb/s). The result of this indication is effectively visible through the MAC register bits *STATUS.SPEED* each time MAC speed has not been forced.
- MAC Dx power state indication: The MAC indicates its ACPI power state (PWR_STATE) to the PHY to enable the PHY to perform intelligent power-management (provided that the PHY power-management is enabled in the MAC CTRL register).
- PHY interrupt: The internal PHY has the capability to generate an interrupt (PHY_INT) on certain conditions, such as link loss events, link status changes, erroneous idle sequences, etc. Configuring the PHY interrupt capability is done through the PHY MII register settings. The interrupt indication from the PHY to the MAC can be enabled as a component-level interrupt, depending on the MAC interrupt-mask IMS settings.

3.6.2.2 MDIO/MDC PHY Management Interface

Foxville implements an IEEE 802.3 MII Management Interface, also known as the Management Data Input/Output (MDIO) or MDIO interface, between the MAC and the internal PHY. This interface provides the MAC and software the ability to monitor and control the state of the PHY. The MDIO interface defines a special protocol that runs across the connection, and an internal set of addressable registers. The interface consists of internal data line (MDIO) and clock line (MDC), which are accessible by software via the MAC register space.

- **Management Data Clock (MDC)**: This signal is used by the internal PHY as a clock timing reference for information transfer on the MDIO signal. The MDC is not required to be a continuous signal and can be frozen when no management data is transferred. The MDC signal has a maximum operating frequency of 2.5 MHz.
- **MDIO**: This bi-directional signal between the MAC and the internal PHY is used to transfer control and status information to and from the PHY (to read and write the PHY management registers).

Software can use MDIO accesses to read or write registers of the internal PHY by accessing Foxville's MDIC register (refer to Section 8.2.4).

3.6.2.2.1 MDIC and MDICNFG Register Usage

For a MDIO read cycle, the sequence of events is as follows:

5. The processor performs a PCIe write cycle to the MDIC register with:
   — Ready = 0b
   — Interrupt Enable set to 1b or 0b
   — Opcode = 10b (read)
   — REGADD = Register address of the specific register to be accessed (0 through 31).
6. The MAC applies the following sequence on the MDIO signal to the PHY:

   <PREAMBLE><01><10><PHYADD=0x0><REGADD><Z> where Z stands for the MAC tri-stating the MDIO signal.
7. The PHY returns the following sequence on the MDIO signal<0><DATA><IDLE>.
8. The MAC discards the leading bit and places the following 16 data bits in the MII register.
9. Foxville asserts an interrupt indicating MDIO Done if the *Interrupt Enable* bit was set.
10. Foxville sets the *Ready* bit in the MDIC register indicating the read completed.
11. The processor might read the data from the MDIC register and issue a new MDIO command.

For a MDIO write cycle, the sequence of events is as follows:

12. The processor performs a PCIe write cycle to the MDIC register with:
    — Ready = 0b.
    — Interrupt Enable set to 1b or 0b.
    — Opcode = 01b (write).
    — REGADD = Register address of the specific register to be accessed (0 through 31).
    — Data = Specific data for desired control of the PHY.
13. The MAC applies the following sequence on the MDIO signal to the PHY:

    <PREAMBLE><01><01><PHYADD=0x0><REGADD><10><DATA><IDLE>
14. Foxville asserts an interrupt indicating MDIO Done if the *Interrupt Enable* bit was set.
15. Foxville sets the *Ready* bit in the MDIC register to indicate that the write operation completed.
16. The CPU might issue a new MDIO command.

Note: A MDIO read or write might take as long as 64 μs from the processor write to the *Ready* bit assertion.

If an invalid opcode is written by software, the MAC does not execute any accesses to the PHY registers.

If the PHY does not generate a 0b as the second bit of the turn-around cycle for reads, the MAC aborts the access, sets the *E* (error) bit, writes 0xFFFF to the data field to indicate an error condition, and sets the *Ready* bit.

Note: After a PHY reset, access through the MDIC register SW should not be attempted for 300 μs.

3.6.2.3 Duplex Operation with Copper PHY

Foxville supports Half-duplex and full-duplex 10/100 Mb/s. Configuring Foxville duplex operation can either be forced or determined via the auto-negotiation process. Refer to Section 3.6.3.1 for details on link configuration setup and resolution.

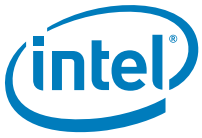
3.6.2.3.1 Full Duplex

All aspects of the IEEE 802.3, 802.3u, 802.3z, and 802.3ab specifications are supported in full-duplex operation. Full-duplex operation is enabled by several mechanisms, depending on the speed configuration of Foxville and the specific capabilities of the link partner used in the application. During full-duplex operation, Foxville can transmit and receive packets simultaneously across the link interface.

In full-duplex, transmission and reception are delineated independently by the GMII/MII control signals. Transmission starts TX_EN is asserted, which indicates there is valid data on the TX_DATA bus driven from the MAC to the PHY/PCS. Reception is signaled by the PHY/PCS by the asserting the RX_DV signal, which indicates valid receive data on the RX_DATA lines to the MAC.

3.6.2.3.2 8B-10B Encoding/Decoding

The GbE PCS circuitry uses the same transmission-coding scheme used in the fiber channel physical layer specification. The 8B10B-coding scheme was chosen by the standards committee in order to provide a balanced, continuous stream with sufficient transition density to allow for clock recovery at the receiving station. There is a 25% overhead for this transmission code, which accounts for the data-signaling rate of 1250 Mb/s with 1000 Mb/s of actual data.



3.6.2.3.3 Code Groups and Ordered Sets

Code group and ordered set definitions are defined in clause 36 of the IEEE 802.3z standard. These represent special symbols used in the encapsulation of GbE packets. The following table contains a brief description of defined ordered sets and included for informational purposes only. Refer to clause 36 of the IEEE 802.3z specification for more details.

Table 3-28. Brief Description of Defined Ordered Sets

Code	Ordered_Set	# of Code Groups	Usage
/C/	Configuration	4	General reference to configuration ordered sets, either /C1/ or /C2/, which is used during auto-negotiation to advertise and negotiate link operation information between link partners. Last 2 code groups contain configuration base and next page registers.
/C1/	Configuration 1	4	See /C/. Differs from /C2/ in 2nd code group for maintaining proper signaling disparity[1].
/C2/	Configuration 2	4	See /C/. Differs from /C1/ in 2nd code group for maintaining proper signaling disparity[1].
/I/	IDLE	2	General reference to idle ordered sets. Idle characters are continually transmitted by the end stations and are replaced by encapsulated packet data.
/I1/	IDLE 1	2	See /I/. Differs from /I2/ in 2nd code group for maintaining proper signaling disparity[1].
/I2/	IDLE 2	2	See /I/. Differs from /I1/ in 2nd code group for maintaining proper signaling disparity[1].
/R/	Carrier_Extend	1	This ordered set is used to indicate carrier extension to the receiving PCS. It is also used as part of the end_of_packet encapsulation delimiter as well as IPG for packets in a burst of packets.
/S/	Start_of_Packet	1	The SPD (start_of_packet delimiter) ordered set is used to indicate the starting boundary of a packet transmission. This symbol replaces the last byte of the preamble received from the MAC layer.
/T/	End_of_Packet	1	The EPD (end_of_packet delimiter) is comprised of three ordered sets. The /T/ symbol is always the first of these and indicates the ending boundary of a packet.
/V/	Error_Propagation	1	The /V/ ordered set is used by the PCS to indicate error propagation between stations. This is normally intended to be used by repeaters to indicate collisions.

1. The concept of running disparity is defined in the standard. In summary, this refers to the 1-0 and 0-1 transitions within 8B10B code groups.

3.6.3 Auto-Negotiation and Link Setup Features

The method for configuring the link between two link partners is highly dependent on the mode of operation as well as the functionality provided by the specific physical layer device (PHY). The PCS and IEEE802.3 clause 28 and clause 40 auto-negotiation functions are maintained within the PHY.

Configuring the link can be accomplished by MAC-controlled auto-negotiation, to auto-negotiation initiated by a PHY. The following sections describe processes of bringing the link up including configuration of Foxville and the transceiver, as well as the various methods of determining duplex and speed configuration.

The PHY performs auto-negotiation per 802.3ab clause 40 and extensions to clause 28. Link resolution is obtained by the MAC from the PHY after the link has been established. The MAC accomplishes this via the MDIO interface, via specific signals from the internal PHY to the MAC, or by MAC auto-detection functions.

3.6.3.1 Copper PHY Link Configuration

When operating with the internal PHY, link configuration is generally determined by PHY auto-negotiation. The software device driver must intervene in cases where a successful link is not negotiated or the designer desires to manually configure the link. The following sections discuss the methods of link configuration for copper PHY operation.

3.6.3.1.1 PHY Auto-Negotiation (Speed, Duplex, Flow Control)

The PHY performs the auto-negotiation function. The actual operational details of this operation are described in the IEEE P802.3ab draft standard and are not included here.

Auto-negotiation provides a method for two link partners to exchange information in a systematic manner in order to establish a link configuration providing the highest common level of functionality supported by both partners. Once configured, the link partners exchange configuration information to resolve link settings such as:

- Speed: - 10/100/1000/2500 Mb/s
- Duplex: - Full or Half (Half duplex is enabled only at 10/100 Mb/s speeds)
- Flow control operation

PHY specific information required for establishing the link is also exchanged.

Note: If flow control is enabled in Foxville, the settings for the desired flow control behavior must be set by software in the PHY registers and auto-negotiation restarted. After auto-negotiation completes, the software device driver must read the PHY registers to determine the resolved flow control behavior of the link and reflect these in the MAC register settings (*CTRL.TFCE* and *CTRL.RFCE*).

Once PHY auto-negotiation completes, the PHY asserts a link indication (LINK) to the MAC. Software must have set the *Set Link Up* bit in the Device Control register (*CTRL.SLU*) before the MAC recognizes the LINK indication from the PHY and can consider the link to be up.

3.6.3.1.2 MAC Speed Resolution

For proper link operation, both the MAC and PHY must be configured for the same speed of link operation. The speed of the link can be determined and set by several methods with Foxville. These include:

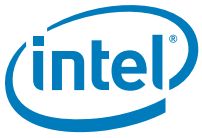
- Software asks the MAC to attempt to auto-detect the PHY speed from the PHY-to-MAC RX_CLK, then programs the MAC speed accordingly
- MAC automatically detects and sets the link speed of the MAC based on PHY indications by using the PHY's internal PHY-to-MAC speed indication (SPD_IND)

Aspects of these methods are discussed in the sections that follow.

3.6.3.1.2.1 Using Internal PHY Direct Link-Speed Indication

Foxville's internal PHY provides a direct internal indication of its speed to the MAC (SPD_IND). When using the internal PHY, the most direct method for determining the PHY link speed and either manually or automatically configuring the MAC speed is based on these direct speed indications.

For MAC speed to be set/determined from these direct internal indications from the PHY, the MAC must be configured such that *CTRL.ASDE* and *CTRL.FRCSPD* are both 0b (both auto-speed detection and forced-speed override disabled). After configuring the Device Control register, MAC speed is re-configured automatically each time the PHY indicates a new link-up event to the MAC.



When MAC speed is neither forced nor auto-sensed by the MAC, the current MAC speed setting and the speed indicated by the PHY is reflected in the Device Status register bits *STATUS.SPEED*.

3.6.3.1.3 MAC Full Duplex Resolution

The duplex configuration of the link is also resolved by the PHY during the auto-negotiation process. Foxville's internal PHY provides an internal indication to the MAC of the resolved duplex configuration using an internal full-duplex indication (FDX).

This internal duplex indication is normally sampled by the MAC each time the PHY indicates the establishment of a good link (LINK indication). The PHY's indicated duplex configuration is applied in the MAC and reflected in the MAC Device Status register (*STATUS.FD*).

Software can override the duplex setting of the MAC via the *CTRL.FD* bit when the *CTRL.FRCDPLX* (force duplex) bit is set. If *CTRL.FRCDPLX* is 0b, the *CTRL.FD* bit is ignored and the PHY's internal duplex indication is applied.

3.6.3.1.4 Using PHY Registers

The software device driver might be required under some circumstances to read from, or write to, the MII management registers in the PHY. These accesses are performed via the MDIC register (refer to Section 8.2.4). The MII registers enable the software device driver to have direct control over the PHY's operation, which can include:

- Resetting the PHY
- Setting preferred link configuration for advertisement during the auto-negotiation process
- Restarting the auto-negotiation process
- Reading auto-negotiation status from the PHY
- Forcing the PHY to a specific link configuration

The set of PHY management registers required for all PHY devices can be found in the IEEE P802.3ab standard.

3.6.3.2 Loss of Signal/Link Status Indication

For all modes of operation, a LOS/LINK signal provides an indication of physical link status to the MAC. This signal from the PHY indicates whether the link is up or down; typically indicated after successful auto-negotiation. Assuming that the MAC has been configured with *CTRL.SLU*=1b, the MAC status bit *STATUS.LU*, when read, generally reflects whether the PHY has link (except under forced-link setup where even the PHY link indication might have been forced).

When the link indication from the PHY is de-asserted the MAC considers this to be a transition to a link-down situation (such as cable unplugged, loss of link partner, etc.). If the Link Status Change (LSC) interrupt is enabled, the MAC generates an interrupt to be serviced by the software device driver.

3.6.4 Ethernet Flow Control (FC)

Foxville supports flow control as defined in 802.3x as well as the specific operation of asymmetrical flow control defined by 802.3z.

Flow control is implemented as a means of reducing the possibility of receive packet buffer overflows, which result in the dropping of received packets, and 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 are defined for the implementation of flow control:

- *CTRL.RFCE* field is used to enable reception of legacy flow control packets and reaction to them
- *CTRL.TFCE* field is used to enable transmission of legacy flow control packets
- Flow Control Address Low, High (FCAL/H) - 6-byte flow control multicast address
- Flow Control Type (FCT) 16-bit field to indicate flow control type
- Flow Control bits in Device Control (CTRL) register - Enables flow control modes
- Discard PAUSE Frames (DPF) and Pass MAC Control Frames (PMCF) in RCTL - controls the forwarding of control packets to the host
- Flow Control Receive Threshold High (FCRTH0) - A 13-bit high watermark indicating receive buffer fullness. A single watermark is used in link FC mode.
- Flow Control Receive Threshold Low (FCRTL0) - A 13-bit low watermark indicating receive buffer emptiness. A single watermark is used in link FC mode.
- Flow Control Transmit Timer Value (FCTTV) - a set of 16-bit timer values to include in transmitted PAUSE frame. A single timer is used in Link FC mode
- Flow Control Refresh Threshold Value (FCRTV) - 16-bit PAUSE refresh threshold value
- RXPBSIZE.Rxpbsize field is used to control the size of the receive packet buffer

3.6.4.1 MAC Control Frames and Receiving Flow Control Packets

3.6.4.1.1 Structure of 802.3X FC Packets

Three comparisons are used to determine the validity of a flow control frame:

1. A match on the 6-byte multicast address for MAC control frames or to the station address of Foxville (Receive Address Register 0).
2. A match on the type field.
3. A comparison of the MAC *Control Op-Code* field.

The 802.3x standard defines the MAC control frame multicast address as 01-80-C2-00-00-01.

The *Type* field in the FC packet is compared against an IEEE reserved value of 0x8808.

The final check for a valid PAUSE frame is the MAC control op-code. At this time only the PAUSE control frame op-code is defined. It 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 quantum (slot time). A pause quantum lasts 64 byte times, which is converted in absolute time duration according to the line speed.

Note: XON frame signals the cancellation of the pause from initiated by an XOFF frame - pause for zero pause quantum.

Table 3-29 lists the structure of a 802.3X FC packet.

Table 3-29. 802.3X Packet Format

DA	01_80_C2_00_00_01 (6 bytes)
SA	Port MAC address (6 bytes)
Type	0x8808 (2 bytes)
Op-code	0x0001 (2 bytes)
Time	XXXX (2 bytes)
Pad	42 bytes
CRC	4 bytes

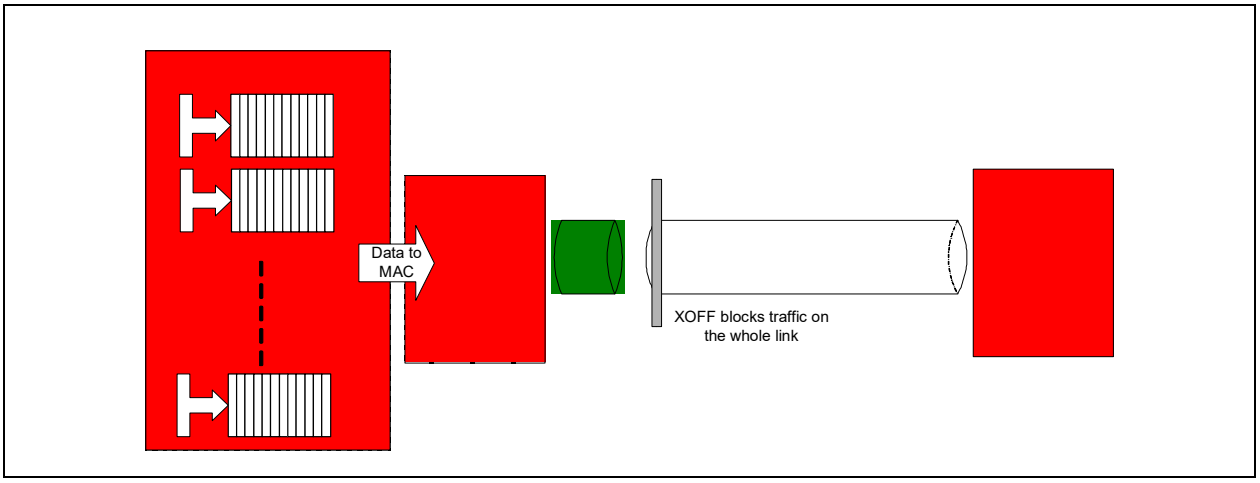


Figure 3-8. MAC Operation

3.6.4.1.2 Operation and Rules

Foxville operates in Link FC.

- Link FC is enabled by the *RFCE* bit in the CTRL register.

Note: Link flow control capability is negotiated between link partners via the auto negotiation process. It is the software device driver responsibility to reconfigure the link flow control configuration after the capabilities to be used where negotiated as it might modify the value of these bits based on the resolved capability between the local device and the link partner.

Once the receiver has validated receiving an XOFF, or PAUSE frame, Foxville performs the following:

- Increments the appropriate statistics register(s)
- Sets the *Flow_Control State* bit in the FCSTS0 register.
- Initializes the pause timer based on the packet's *PAUSE* timer field (overwriting any current timer's value)
- Disables packet transmission or schedules the disabling of transmission after the current packet completes.

Resumption of transmission might occur under the following conditions:

- Expiration of the PAUSE timer
- Receiving an XON frame (a frame with its PAUSE timer set to 0b)

Both conditions clear the relevant *Flow_Control State* bit in the relevant FCSTS0 register and transmission can resume. Hardware records the number of received XON frames.

3.6.4.1.3 Timing Considerations

When operating at 1 Gb/s line speed, Foxville must not begin to transmit a (new) frame more than two pause-quantum-bit times after receiving a valid link XOFF frame, as measured at the wires. A pause quantum is 512-bit times.

When operating in full duplex at 10 Mb/s or at 100 Mb/s line speeds, Foxville must not begin to transmit a (new) frame more than 576-bit times after receiving a valid link XOFF frame, as measured at the wire.

3.6.4.2 PAUSE and MAC Control Frames Forwarding

Two bits in the Receive Control register, control forwarding of PAUSE and MAC control frames to the host. These bits are *Discard PAUSE Frames* (*DPF*) and *Pass MAC Control Frames* (*PMCF*):

- The *DPF* bit controls forwarding of PAUSE packets to the host.
- The *PMCF* bit controls forwarding of non-PAUSE packets to the host.

Note: When flow control reception is disabled (*CTRL.RFCE* = 0b), legacy flow control packets are not recognized and are parsed as regular packets.

Table 3-30 lists the behavior of the *DPF* bit.

Table 3-30. Forwarding of PAUSE Packet to Host (*DPF* Bit)

RFCE	DPF	Are FC Packets Forwarded to Host?
0	X	Yes if pass the L2 filters (refer to Section 7.1.1.1).[1]
1	0	Yes.
1	1	No.

1. The flow control multicast address is not part of the L2 filtering unless explicitly required.

Table 3-31 defines the behavior of the *PMCF* bit.

Table 3-31. Transfer of Non-PAUSE Control Packets to Host (*PMCF* Bit)

RFCE	PMCF	Are Non-FC MAC Control Packets Forwarded to Host?
0	X	Yes if pass the L2 filters (refer to Section 7.1.1.1).
1	1	Yes.
1	0	No.

3.6.4.3 Transmission of PAUSE Frames

Foxville generates PAUSE packets to ensure there is enough space in its receive packet buffers to avoid packet drop. Foxville monitors the fullness of its receive packet buffers and compares it with the contents of a programmable threshold. When the threshold is reached, Foxville sends a PAUSE frame. Foxville supports the sending of link Flow Control (FC).

Note: Similar to receiving link flow control packets previously mentioned, link XOFF packets can be transmitted only if this configuration has been negotiated between the link partners via the auto-negotiation process or some higher level protocol. The setting of this bit by the software device driver indicates the desired configuration.

3.6.4.3.1 Operation and Rules

Transmission of link PAUSE frames is enabled by software writing a 1b to the *TFCE* bit in the Device Control register.

Foxville sends a PAUSE frame when Rx packet buffer is full above the high threshold defined in the Flow Control Receive Threshold High (*FCRTH0.RTH*) register field. When the threshold is reached, Foxville sends a PAUSE frame with its pause time field equal to *FCTTV*. The threshold should be large enough to overcome the worst case latency from the time that crossing the threshold is sensed until packets are not received from the link partner. The Flow Control Receive Threshold High value should be calculated as follows:

Flow Control Receive Threshold High = Internal Rx Buffer Size - (Threshold Cross to XOFF Transmission + Round-trip Latency + XOFF Reception to Link Partner response)

Parameter values to be used for calculating the *FCRTH0.RTH* value are listed in Table 3-32.

Table 3-32. Flow Control Receive Threshold High (*FCRTH0.RTH*) Value Calculation

Latency Parameter	Affected by	Parameter Value
Internal Rx Buffer Size	Internal Tx buffer size.	60 KB - Internal Tx Buffer Size.
Threshold Cross to XOFF Transmission[1]	Max packet size.	Max packet size * 1.25.
XOFF Reception to Link Partner response	Max packet size.	Max packet size.
Round trip latency[2]	The latencies on the wire and the LAN devices at both sides of the wire.	320-byte for 1000Base-T operation. 800 bytes for 2.5 Gb/s operation

1. 1.25 multiplier added to address internal RX packet overhead (128 bit time stamp + 64 bit descriptor) when receiving 65 Byte packets while transmitting 9.5 KB Jumbo frame, with some additional margin.
2. Includes 2 x pause quanta on RX of 1024 bit time, Tx GMII to MDI delay of 84 bit time, 2 * max link delay of 1140 bit time plus 29 byte time for processing margin.

After transmitting a PAUSE frame, Foxville activates an internal shadow counter that reflects the link partner pause timeout counter. When the counter reaches the value indicated in the FCRTV register, then, if the PAUSE condition is still valid (meaning that the buffer fullness is still above the high watermark), a XOFF message is sent again.

Once the receive buffer fullness reaches the low water mark, Foxville sends a XON message (a PAUSE frame with a timer value of zero). Software enables this capability with the *XONE* field of FCRTL.

Foxville sends an additional PAUSE frame if it has previously sent one and the packet buffer overflows. This is intended to minimize the amount of packets dropped if the first PAUSE frame did not reach its target.

3.6.4.3.2 Software Initiated PAUSE Frame Transmission

Foxville has the added capability to transmit an XOFF frame via software. This is accomplished by software writing a 1b to the *SWXOFF* bit of the Transmit Control register. Once this bit is set, hardware initiates the transmission of a PAUSE frame in a manner similar to that automatically generated by hardware.

The *SWXOFF* bit is self-clearing after the PAUSE frame has been transmitted.

Note: The Flow Control Refresh Threshold mechanism does not work in the case of software-initiated flow control. Therefore, it is the software's responsibility to re-generate PAUSE frames before expiration of the pause counter at the other partner's end.

The state of the *CTRL.TFCE* bit or the negotiated flow control configuration does not affect software generated PAUSE frame transmission.

Note: Software sends an XON frame by programming a 0b in the PAUSE timer field of the FCTTV register. Software generating an XON packet is not allowed while the hardware flow control mechanism is active, as both use the FCTTV registers for different purposes.

When flow control is disabled, pause packets (XON, XOFF, and other FC) are not detected as flow control packets and can be counted in a variety of counters (such as multicast).

3.6.4.4 IPG Control and Pacing

Foxville supports the following modes of controlling IPG duration:

- Fixed IPG - the IPG is extended by a fixed duration

3.6.4.4.1 Fixed IPG Extension

Foxville allows controlling of the IPG duration. The IPGT configuration field enables an extension of IPG in 4-byte increments. One possible use of this capability is to enable inserting bytes into the transmit packet after it has been transmitted by Foxville without violating the minimum IPG requirements. For example, a security device connected in series to Foxville might add security headers to transmit packets before the packets are transmitted on the network.

3.6.5 Loopback Support

3.6.5.1 General

Foxville supports the following types of internal loopback in the LAN interface:

- MAC Loopback (Point 1)
- PHY Loopback (Point 2)
- External Loopback (point 3)

By setting the device to loopback mode, packets that are transmitted towards the line are looped back to the host. Foxville is fully functional in these modes, just not transmitting data over the lines. Figure 3-9 shows the points of loopback.

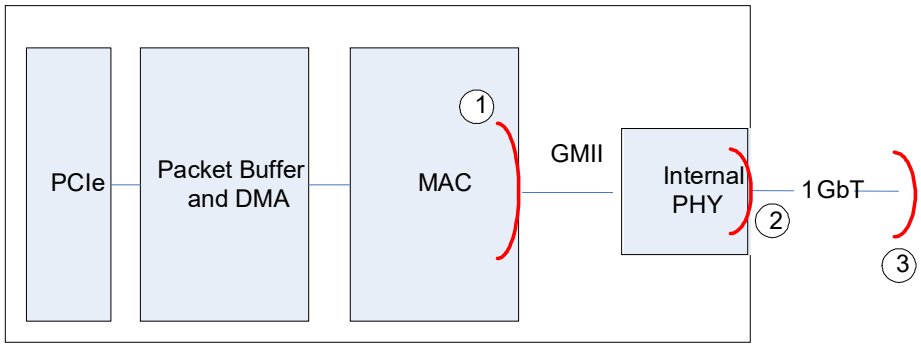


Figure 3-9. Foxville Loopback Modes

In addition, Foxville's copper PHY support a far end loopback mode, where incoming traffic is reflected at the PHy level onto the transmit wires. This mode is entered by setting bit 14 in PHY register Page 2, Register 21.

3.6.5.2 Generic Loopback Flow

3.6.5.2.1 Entering the Loopback Flow

- All loopback modes are intended for debug rather than nominal operation. Some of the loopback setups are not reflected explicit in device registers.

3.6.5.2.2 Exiting the Loopback Flow

- Going back to nominal transmit / receive operation the software sets the device to nominal transmit / receive.

3.6.5.3 MAC Loopback

In MAC loopback, the PHYblocks are not functional and data is looped back before these blocks.

3.6.5.3.1 Setting Foxville to MAC loopback Mode

The following procedure should be used to put Foxville in MAC loopback mode:

- Set *RCTL.LBM* to 01b (bits 7:6)
- Set *CTRL.SLU* (bit 6, should be set by default)
- Set *CTRL.FRCSPD* and *FRCDPLX* (bits 11 and 12)
- Set the *CTRL.FD* bit and program the *CTRL.SPEED* field to 110b (2.5 GbE).
- Set *EEER.EEE_FRC_AN* to 1b to enable checking EEE operation in MAC loopback mode.

Filter configuration and other Tx/Rx processes are the same as in normal mode.

3.6.5.4 PHY Loopback

In PHY loopback, the data is looped back at the end of the PHY functionality. This means all the design, that is functional in copper mode, is involved in the loopback.

3.6.5.4.1 Setting Foxville to Internal PHY loopback Mode

The following procedure should be used to place Foxville in PHY loopback mode on LAN port:

- In the PHY control register (*PHYREG 0,0* - Address 0 in the PHY):
  - Set duplex mode (bit 8)
  - Clear auto-negotiation enable bit (bit 12)
  - Set speed using bits 6 and 13as described in EAS. Register values should be:
    - For 10 Mb/s 0x4100
    - For 100 Mb/s 0x6100
    - For 1000 Mb/s 0x4140
    - For 2.5 Gb/s 0x6140
  - Use bits 2:0 in PHYREG 2,21 to control the link speed in MDI loopback
  - reset the PHY – in PHYREG 0,0 Set Copper Reset bit (bit 15)
  - In PHYREG 0,0 Set loopback bit (bit 14)

3.6.5.4.2 Setting Foxville Internal PHY to External Loopback Mode

The following procedures should be used to put Foxville internal PHY into external loopback mode:

Loopback at 2.5Gbps
1. PHY Register 7.32 = 0x40A2 // Enable 2.5G in the ANEG_MGBT_AN_CTRL register
2. PHY Register 0.19 = 0x8000 // Disable auto-MDIX + enable RJ45 loop in the PHY_CTL1 register
3. PHY Register 0.0 = 0x1200 // Restart ANEG in the STD_CTRL register

Loopback at 1Gbps
1. PHY Register 7.32 = 0x4002 // Disable 2.5G in the ANEG_MGBT_AN_CTRL register
2. PHY Register 7.62 = 0x0 // Disable EEE in the ANEG_EEE_AN_ADV2 register
3. PHY Register 0.19 = 0x8000 // Disable auto-MDIX + enable RJ45 loop in the PHY_CTL1 register
4. PHY Register 0.0 = 0x1200 // Restart ANEG in the STD_CTRL register

Loopback at 100Mbps
1. PHY Register 7.32 = 0x4002 // Disable 2.5G in the ANEG_MGBT_AN_CTRL register
2. PHY Register 7.62 = 0x0 // Disable EEE in the ANEG_EEE_AN_ADV2 register
3. PHY Register 0.9 = 0x0 // Disable 1G in the STD_GCTRL register
4. PHY Register 0.19 = 0x8000 // Disable auto-MDIX + enable RJ45 loop in the PHY_CTL1 register
5. PHY Register 0.0 = 0x1200 // Restart ANEG in the STD_CTRL register

Loopback at 10Mbps
1. PHY Register 7.32 = 0x4002 // Disable 2.5G in the ANEG_MGBT_AN_CTRL register
2. PHY Register 7.62 = 0x0 // Disable EEE in the ANEG_EEE_AN_ADV2 register
3. PHY Register 0.9 = 0x0 // Disable 1G in the STD_GCTRL register
4. PHY Register 0.4 = 0x0061 // Disable 100M in the STD_AN_ADV register
5. PHY Register 0.19 = 0x8000 // Disable auto-MDIX + enable RJ45 loop in the PHY_CTL1 register
6. PHY Register 0.0 = 0x1200 // Restart ANEG in the STD_CTRL register

3.6.5.5 Line Loopback

In line loopback (Figure 3-10), MAC interfaces are not functional and the data is sent from a link partner to the PHY to test transmit and receive data paths. Frames that originate from a link partner are looped back from the PHY and sent out on the wire before reaching the MAC interface pins.

The following should be confirmed before enabling the line loopback feature:

- The PHY must first establish a full-duplex link with another PHY link partner, either through auto-negotiation or through forcing the same link speed.

In order to enable line loopback mode once the link is established, set bit 14 to 1b in the MAC Specific Control Register 2 - Page 2, Register 21 in the PHY.

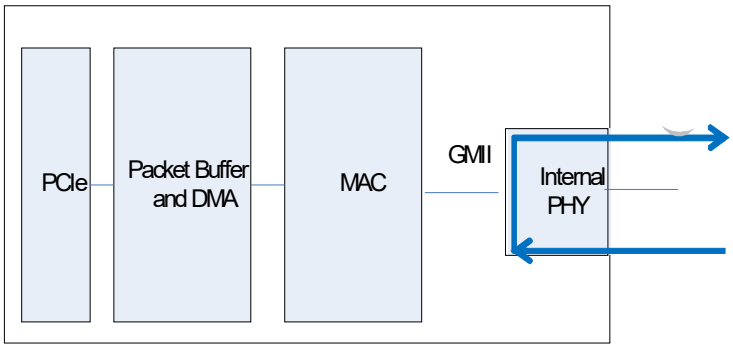


Figure 3-10. Line Loopback

3.6.6 Energy Efficient Ethernet (EEE)

Energy Efficient Ethernet (EEE) Low Power Idle (LPI) mode defined in IEEE802.3az optionally enables power saving by switching off part of Foxville functionality when no data needs to be transmitted or/ and received. The decision as to whether or not Foxville transmit path should enter LPI mode or exit LPI mode is done according to transmit needs. Information as to whether or not a link partner has entered LPI mode is detected by Foxville and is 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 PHY to transmit sleep symbols for a pre-defined period of time followed by a quite period. During LPI, the 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 transmitting normal inter-frame encoding on the internal xxMII interface. The PHY communicates to the link partner the move to active link state by sending wake symbols for a pre-defined 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 a link partner enters LPI, the PHY indicates assert low power idle on the internal xxMII RX interface and Foxville's receiver disables certain functionality to reduce power consumption.

Figure 3-11 shows and Table 3-33 lists the general principles of EEE LPI operation on the Ethernet Link.

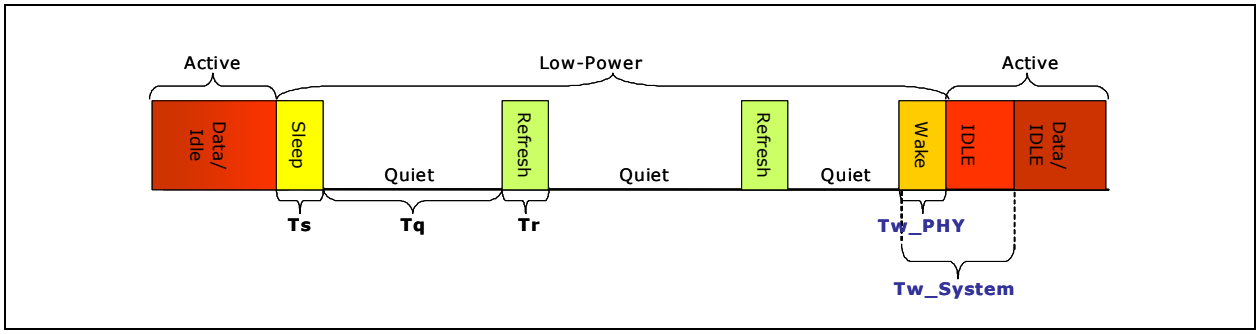


Figure 3-11. Energy Efficient Ethernet Operation

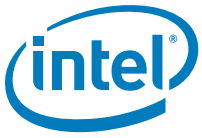
Table 3-33. Energy Efficient Ethernet Parameters

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 an 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.
Transmit System Wake Time (Tw_System_tx)	Wait period where no data is transmitted to give the remote receiving system time to wake up.

3.6.6.1 Conditions to Enter EEE Tx LPI

Following link-up or after Auto-negotiation Foxville will not enter TX LPI for the duration defined in the *EEE_SU.TX_LU_LPI_DLY* field + PHY_LPI_DLY. The PHY_LPI_DLY is set according to the PHY image in the NVM to 2 sec at 2.5Gb/s and 5 sec at 1Gb/s and 100Mb/s. During this time the link is up and transmission is enable. Then, Entry into EEE LPI mode of operation is triggered when one of the following conditions exist:

1. EEE is enabled by the *EEER.TX_LPI_EN* and no transmission is pending from both the host as well as the manageability (both transmit buffers are empty) for as long as defined by the *EEE_SU.TEEE_DLY.* The *EEE_SU.TEEE_DLY* parameter is set to avoid frequent entry into TX LPI state for short durations.
2. If the *EEER.TX_LPI_EN* and *EEER.LPI_FC* bits are set to 1b and a XOFF flow control packet is received from the link partner, Foxville moves the link into the Tx LPI state for the pause duration even if a transmission is pending.
3. When *EEER.Force_TLPI* is set (even if *EEER.TX_LPI_EN* is cleared).



— If *EEER.Force_TLPI* is set in mid-packet, Foxville completes packet transmission and then moves Tx to LPI.

— Setting the *EEER.Force_TLPI* bit to 1b only stops transmission of packets from the host. Foxville moves the link out of Tx LPI to transmit packets from management even when *EEER.Force_TLPI* is set to 1b.

4. When a function enters D3 state and there's no management Tx trafficand internal transmit buffers are empty and *EEER.TX_LPI_EN* is set to 1b.

When one of the previous conditions to enter a Tx LPI state is detected, assert low power idle is transmitted on the internal xxMII interface and Foxville PHY transmits sleep symbols on the network interface to communicate to the link partner entry into Tx LPI link state. After sleep symbols transmission, behavior of the PHY differs according to link speed (100BASE-TX or 1000BASE-T or 2.5 Gb/s):

1. While in 100BASE-TX mode, the PHY enters low power operation in an asymmetric manner. After sleep symbol transmissions, the PHY immediately enters a low power quiet state.
2. While in 1000BASE-T mode and 2.5 Gb/s mode, the PHY entry into a quiet state is symmetric. Only after the PHY transmits sleep symbols and receives sleep symbols from the remote PHY does the PHY enter the quiet state.

After entering a quiet link state, the PHY periodically transitions between quiet link state, where link is idle, to sending refresh symbols until a request to transition the link back to normal (active) mode is transmitted on the internal xxMII TX interface (see Figure 3-11).

Note: MAC entry into Tx LPI state is always asymmetric (in all link speeds).

3.6.6.2 Exit of TX LPI to Active Link State

Foxville exits Tx LPI link state and transition link into active link state when none of the conditions defined in Section 3.6.6.1 exist. Transitioning into active link state, Foxville transmits:

1. Normal inter-frame encoding on the internal xxMII TX interface for a pre-defined link rate dependant period time of Tw_sys_tx-min (defined in the *EEE_SU.Tw_min_1000* and *EEE_SU.Tw_min_100* fields). As a result, PHY transmits wake symbols for a Tw_phy duration followed by idle symbols.
2. If the Tw_System_tx duration defined in the *EEER.Tw_system* field is longer than Tw_sys_tx-min, Foxville continues transmitting the inter-frame encoding on the internal xxMII interface until the time defined in the *EEER.Tw_system* field has expired, before transmitting the actual data. During this period the PHY continues transmitting idle symbols.

Note: When moving out of Tx LPI to transmit a 802.3x flow control frame Foxville waits only the Tw_sys_tx-min 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 *EEER.Tw_system* field has expired.

Note: Once TX LPI has been entered Foxville will not initiate transition of the port back to an active state until at least duration defined in the *EEE_SU.Tw_wake_min* field has passed (even if above conditions to exit TX LPI exist).

3.6.6.3 EEE Auto-Negotiation

Auto-negotiation provides the capability to negotiate EEE capabilities with the link partner using the next page mechanism defined in IEEE802.3 Annex 28C. IEEE802.3 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 via the IEEE802.3 auto-negotiation process. If EEE is supported by both link partners for the negotiated PHY type then the EEE function can be used independently in either direction.

Foxville supports EEE auto-negotiation. EEE capabilities advertised during auto-negotiation can be modified via the *EEE_2_5G_AN ; EEE_1G_AN* and *EEE_100M_AN* flags in the IPCNFG register.

3.6.6.4 EEE Link Level (LLDP) Capabilities Discovery

Foxville supports LLDP negotiation via software, using the EEE IEEE802.1AB Link Layer Discovery Protocol (LLDP) Type, Length, Value (TLV) fields defined in IEEE802.3az 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.

After software negotiates a new system wake time via EEE LLDP negotiation, software should update the:

1. EEER.Tw_system field with the negotiated Transmit T_w time value, to increase the duration where idle symbols are transmitted following move out of EEE Tx LPI state before actual data can be transmitted.
   — Value placed in *EEER.Tw_system* field does not affect transmission of flow control packets. Depending on the technology (100BASE-TX or 1000BASE-T or 2500BASE-T) flow control packet transmission is delayed following move out of EEE TX LPI state only by the minimum Tw_sys_tx time as defined in IEEE802.3az clause 78.5. In Foxville the minimum Tw_sys_tx time value is defined in the *EEE_SU* register. Value varies as a function of link rate (100BASE-TX or 1000BASE-T or 2500BASE-T).
2. The LTRMAXV register with a value:

   LTRMINV =< *LTRMAXV* <= *LTRMINV* + negotiated Receive T_w Time.

3. Set *LTRC.EEEMS_EN* bit to 1b (if bit was cleared), so that on detection of EEE RX LPI on the network an updated LTR message with the value programmed in the LTRMAXV register is sent on the PCIe interface.
4. Set *EEER.TX_LPI_EN* bit to 1b (if bit was cleared), to enable entry into EEE LPI on Tx path.
5. Set *EEER.RX_LPI_EN* bit to 1b (if bit was cleared), to enable detection of link partner entering EEE LPI state on Rx path.

Once the *LTRC.EEEMS_EN* bit is set and a port detects link partner entry into the EEE LPI state on the internal xxMII RX interface, the port increases its reported latency tolerance to the value programed in the LTRMAXV register. On detection of the Rx EEE LPI state, an updated LTR message is sent on the PCIe interface.

When wake symbols are detected on the Ethernet link, due to a link partner moving out of EEE Rx LPI state, the port reports a reduced latency tolerance that equals the value placed in the LTRMINV register and Foxville sends on the PCIe interface a new LTR message with a reduced latency tolerance value of *LTRMINV.*

Note: If link is disconnected or auto-negotiation is re-initiated, then the *LTRC.EEEMS_EN* bit is cleared by hardware. The bit should be set to 1b by software following re-execution of an EEE LLDP negotiation.

Figure 3-12 shows the format of the EEE TLV, meaning of the various TLV parameters can be found in IEEE802.3az clause 78 and clause 79.

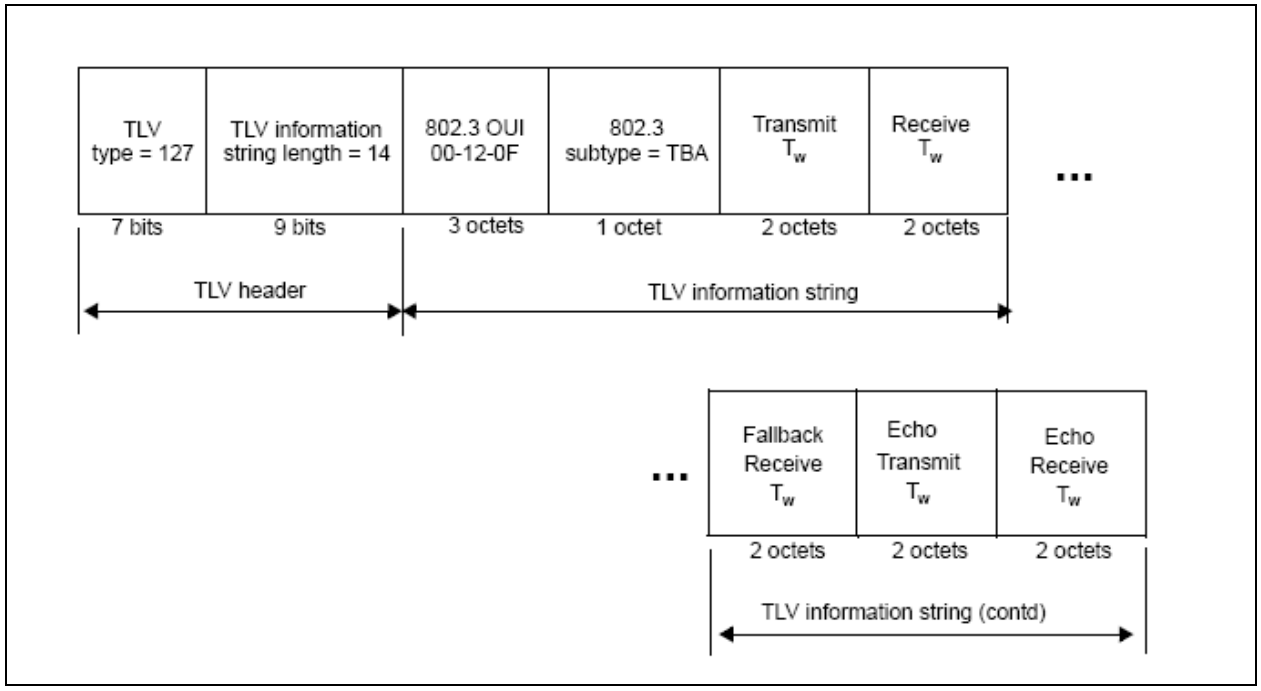


Figure 3-12. EEE LLDP TLV

3.6.6.5 Programming Foxville for EEE Operation

To activate EEE support, software should program the following fields to enable EEE on LAN port:

1. IPCNFG register (refer to Section 8.25.1) if default EEE advertised auto-negotiation values need to be modified.
2. Set the *EEER.TX_LPI_EN and EEER.RX_LPI_EN* bits (refer to Section 8.24.5) to 1b to enable EEE LPI support on Tx and Rx paths, respectively, if the result of auto-negotiation at the specified link speed enables entry to LPI.
3. Set the *EEER.LPI_FC* bit (refer to Section 8.24.5) if required to enable a move into the EEE Tx LPI state for the pause duration when a link partner sends a XOFF flow control packet even if internal transmit buffer is not empty and transmit descriptors are available.
4. Update *EEER.Tw_system* field (refer to Section 8.24.5) with the new negotiated transmit T_w time after completing EEE LLDP negotiation.
5. Modify *EEE_SU.TEEE_DLY* field (refer to Section 8.24.6) if delay before deciding to enter TX EEE LPI state needs to be modified.
6. Modify other *EEE_SU* (refer to Section 8.24.6) register field values if other default values need to be modified.
7. Following the EEE LLDP negotiation program, the LTRMAXV register (refer to Section 8.24.3) with a value of:

   LTRMINV =< *LTRMAXV* <= *LTRMINV* + negotiated Receive T_w Time.

8. Set the *LTRC.EEEMS_EN* bit to 1b, to enable sending an updated PCIe LTR message when detecting a link partner entry into EEE Rx LPI state.

Notes:

1. The *LTRC.EEEMS_EN* bit is cleared following link disconnect or auto-negotiation and should be set to 1b by software following EEE LLDP re-negotiation.
2. Foxville waits for at least 1 second (depending on value in *EEE_SU.TX_LPI_DLY* field) following auto-negotiation (due to reset, link disconnect, or link speed change) and link-up indication (*STATUS.LU* set to 1b, refer to Section 8.2.2) before enabling link entry into EEE Tx LPI state to comply with the IEEE802.3az specification.

3.6.6.6 EEE Statistics

Foxville supports reporting the number of EEE LPI Tx and Rx events via the RLPIC and TLPIC registers.

3.6.7 Integrated Copper PHY Functionality

3.6.7.1 Interconnects of the Integrated PHY

The integrated PHY is connected to the MAC unit by the following signals:

- Transmit and Receive data bus
- Out of band MDIO signals
- Dedicated Input / Output Control signals

3.6.7.1.1 Transmit and Receive data bus - GMII

Transmit and receive data as well as EEE related messages are transferred between the MAC and the PHY on an internal standard GMII bus. The data bus is 8 bits wide. And it is extended to 2.5 Gb/s operation.

3.6.7.1.2 Out of band MDIO signals

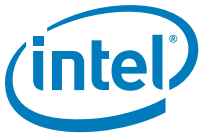
Software as well as the embedded management controller can access registers in the integrated PHY unit by the out of band MDIO signals. Driving the MDIO signals are done via programming of the MDIC and MDICNFG registers. Note that the MDICNFG registers is loaded from the NVM not expected to be modified by the software nor the embedded management controller. See Section 3.6.2.2 for additional description of the flow accessing the PHY registers.

3.6.7.1.3 1588 related Signals

The PHY indicates the transmission and reception of transmit and receive packets on dedicated signals to the MAC cluster. The TX_SOP indicates the beginning of a transmit packet and the RX_SOP indicates the beginning of a receive packet.

3.6.7.1.4 Dedicated Input / Output Control signals

On top of the option to control the PHY functionality, the MAC has direct connection to some critical signals in the PHY.



3.6.7.2 Determining Link State

The PHY and its link partner determine the type of link established through one of three methods:

- Auto-negotiation
- Parallel detection
- Forced operation (not supported)

Auto-negotiation is the only method allowed by the 802.3ab standard for establishing a 2500BASE-T link or 1000BASE-T link, although forced operation could be used for test purposes. For 10/100 links, any of the three methods can be used. The following sections discuss each in greater detail.

Figure 3-13 provides an overview of link establishment. First the PHY checks if auto-negotiation is enabled. By default, the PHY supports auto-negotiation, see PHY Register 0, bit 12. If not, the PHY forces operation as directed. If auto-negotiation is enabled, the PHY begins transmitting Fast Link Pulses (FLPs) and receiving FLPs from its link partner. If FLPs are received by the PHY, auto-negotiation proceeds. It also can receive 100BASE-TX MLT3 and 10BASE-T Normal Link Pulses (NLPs).

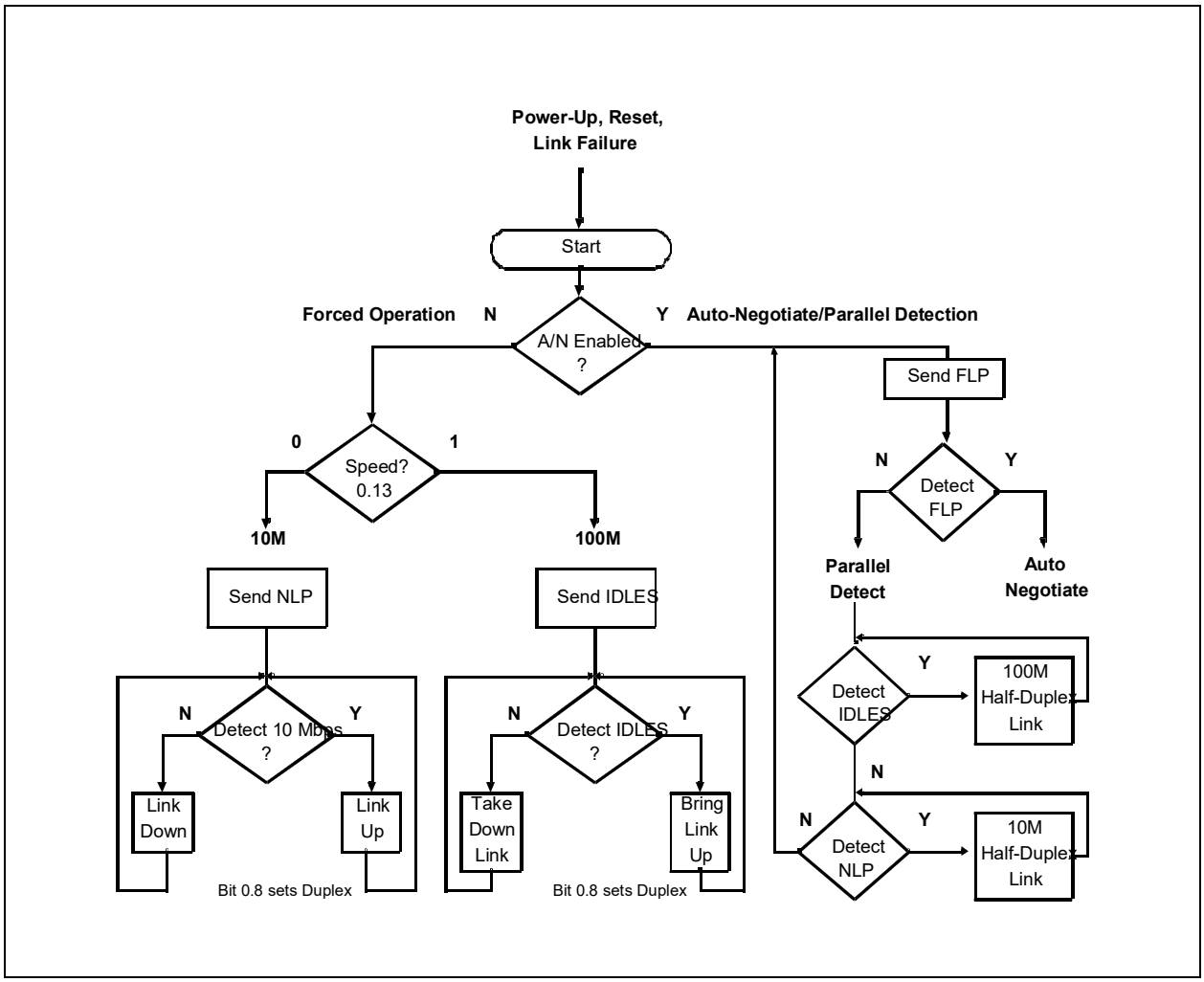


Figure 3-13. Overview of Link Establishment

3.6.7.2.1 Auto Negotiation

The PHY supports the IEEE 802.3u auto-negotiation scheme with next page capability. Next page exchange uses Register 7 to send information and Register 8 to receive them. Next page exchange can only occur if both ends of the link advertise their ability to exchange next pages.

3.6.7.2.2 Parallel Detection

Parallel detection can only be used to establish 10 and 100 Mb/s links. It occurs when the PHY tries to negotiate (transmit FLPs to its link partner), but instead of sensing FLPs from the link partner, it senses 100BASE-TX MLT3 code or 10BASE-T Normal Link Pulses (NLPs) instead. In this case, the PHY immediately stops auto-negotiation (terminates transmission of FLPs) and immediately brings up whatever link corresponds to what it has sensed (MLT3 or NLPs). If the PHY senses both technologies, the parallel detection fault is detected and the PHY continues sending FLPs.

Parallel detection also does not allow exchange of flow-control ability (PAUSE and ASM_DIR) or the master/slave relationship required by 1000BASE-T and 2500BASE-T. This is why parallel detection cannot be used to establish GbE links.

3.6.7.2.3 Auto Cross-Over

Twisted pair Ethernet PHY's must be correctly configured for MDI or MDI-X operation to inter operate. This has historically been accomplished using special patch cables, magnetics pinouts or Printed Circuit Board (PCB) wiring. The PHY supports the automatic MDI/MDI-X configuration originally developed for 1000Base-T and standardized in IEEE 802.3u section 40. Manual (non-automatic) configuration is still possible.

For 1000BASE-T and 2500BASE-T links, pair identification is determined automatically in accordance with the standard.

For 10/100/1000/2500 Mb/s links and during auto-negotiation, pair usage is determined by bits 4 and 5 in PHYREG 0,21. The PHY activates an automatic cross-over detection function if enabled via bit 0 in IPCNFG (also see bits 5 and 6 in PHYREG 0,16). When in this mode, the PHY automatically detects which application is being used and configures itself accordingly.

The automatic MDI/MDI-X state machine facilitates switching the MDI_PLUS[0] and MDI_MINUS[0] signals with the MDI_PLUS[1] and MDI_MINUS[1] signals, respectively, prior to the auto-negotiation mode of operation so that FLPs can be transmitted and received in compliance with Clause 28 auto-negotiation specifications. An algorithm that controls the switching function determines the correct polarization of the cross-over circuit. This algorithm uses an 11-Bit Linear Feedback Shift Register (LFSR) to create a pseudo-random sequence that each end of the link uses to determine its proposed configuration. After making the selection to either MDI or MDI-X, the node waits for a specified amount of time while evaluating its receive channel to determine whether the other end of the link is sending link pulses or PHY-dependent data. If link pulses or PHY-dependent data are detected, it remains in that configuration. If link pulses or PHY-dependent data are not detected, it increments its LFSR and makes a decision to switch based on the value of the next bit. The state machine does not move from one state to another while link pulses are being transmitted.

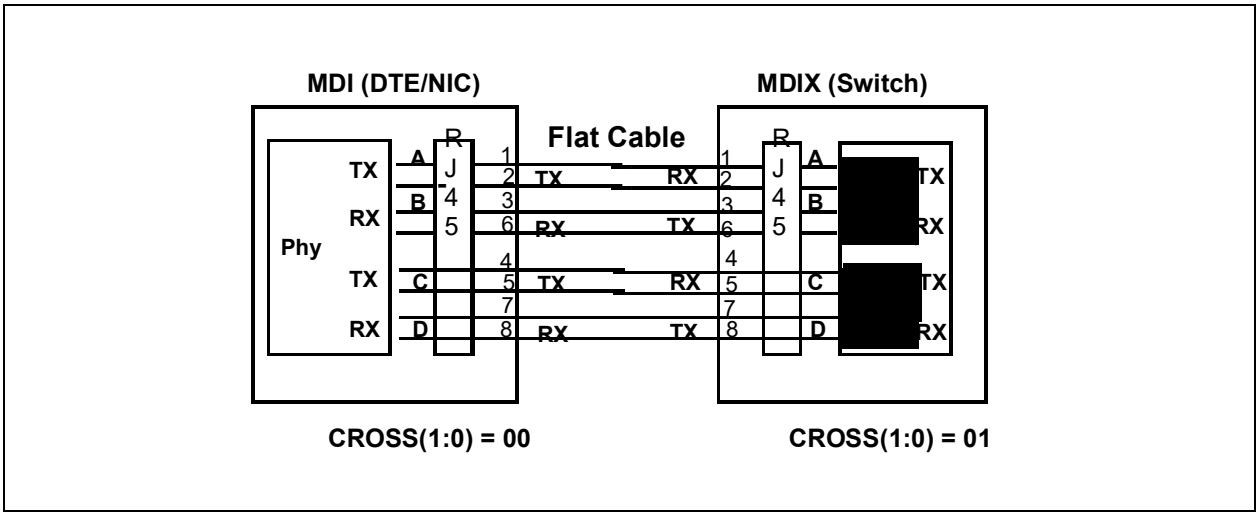


Figure 3-14. Cross-Over Function

3.6.7.2.4 10/100 MB/s Mismatch Resolution

It is a common occurrence that a link partner (such as a switch) is configured for forced full-duplex (FDX) 10/100 Mb/s operation. The normal auto-negotiation sequence would result in the other end settling for half-duplex (HDX) 10/100 Mb/s operation. The mechanism described in this section resolves the mismatch automatically and transitions Foxville into FDX mode, enabling it to operate with a partner configured for FDX operation.

Foxville enables the system software device driver to detect the mismatch event previously described and sets its duplex mode to the appropriate value without a need to go through another auto-negotiation sequence or breaking link. Once software detects a possible mismatch, it might instruct Foxville to change its duplex setting to either HDX or FDX mode. Software sets the *Duplex_manual_set* bit to indicate that duplex setting should be changed to the value indicated by the *Duplex Mode* bit in PHY Register 0. Any change in the value of the *Duplex Mode* bit in PHY Register 0 while the *Duplex_manual_set* bit is set to 1b would also cause a change in the device duplex setting.

The *Duplex_manual_set* bit is cleared on all PHY resets, following auto-negotiation, and when the link goes down. Software might track the change in duplex through the PHY *Duplex Mode* bit in Register 17 or a MAC indication.

3.6.7.2.5 Link Criteria

Once the link state is determined-via auto-negotiation, parallel detection or forced operation, the PHY and its link partner bring up the link.

3.6.7.2.5.1 1000BASE-T / 2500BASE-T

For 1000BASE-T / 2500BASE-T links, the PHY and its link partner enter a training phase. They exchange idle symbols and use the information gained to set their adaptive filter coefficients. These coefficients are used to equalize the incoming signal, as well as eliminate signal impairments such as echo and cross talk.

Either side indicates completion of the training phase to its link partner by changing the encoding of the idle symbols it transmits. When both sides so indicate, the link is up. Each side continues sending idle symbols each time it has no data to transmit. The link is maintained as long as valid idle, data, or carrier extension symbols are received.

3.6.7.2.5.2 100BASE-TX

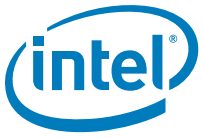
For 100BASE-TX links, the PHY and its link partner immediately begin transmitting idle symbols. Each side continues sending idle symbols each time it has no data to transmit. The link is maintained as long as valid idle symbols or data is received.

In 100 Mb/s mode, the PHY establishes a link each time the scrambler becomes locked and remains locked for approximately 50 ms. Link remains up unless the descrambler receives less than 12 consecutive idle symbols in any 2 ms period. This provides for a very robust operation, essentially filtering out any small noise hits that might otherwise disrupt the link.

3.6.7.2.5.3 10BASE-Te

For 10BASE-Te links, the PHY and its link partner begin exchanging Normal Link Pulses (NLPs). The PHY transmits an NLP every 16 ms and expects to receive one every 10 to 20 ms. The link is maintained as long as normal link pulses are received.

In 10 Mb/s mode, the PHY establishes link based on the link state machine found in 802.3, clause 14.



Note: 100 Mb/s idle patterns do not bring up a 10 Mb/s link.

3.6.7.3 SmartSpeed (Downspeed)

SmartSpeed (Downspeed) is an enhancement to auto-negotiation that enables the PHY to react intelligently to network conditions that prohibit establishment of a 1000BASE-T or 2500BASE-T links, such as cable problems. Such problems might allow auto-negotiation to complete, but then inhibit completion of the training phase.

3.6.7.3.1 Using SmartSpeed

When Downspeed downgrades the PHY advertised capabilities, it sets bit *Downspeed Status*. When link is established, its speed is indicated in the *Speed* field. Downspeed automatically resets the highest-level auto-negotiation abilities advertised, if link is established and then lost.

Note: Time To Link (TTL) with Downspeed - in most cases, any attempt duration is approximately 2.5 seconds, in other cases it could take more than 2.5 seconds depending on configuration and other factors.

3.6.7.4 Flow Control

Flow control is a function that is described in Clause 31 of the IEEE 802.3 standard. It enables congested nodes to pause traffic. Flow control is essentially a MAC-to-MAC function. MACs indicate their ability to implement flow control during auto-negotiation. This ability is communicated through two bits in the auto-negotiation registers (*PHYREG 0,4.10* and *PHYREG 0,4.11*).

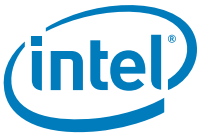
The PHY transparently supports MAC-to-MAC advertisement of flow control through its auto-negotiation process. Prior to auto-negotiation, the MAC indicates its flow control capabilities via *PHYREG 0,4.10* (Pause) and *PHYREG 0,4.11* (*ASM_DIR*). After auto-negotiation, the link partner's flow control capabilities are indicated in *PHYREG 0,5.10* and *PHYREG 0,5.11*.

There are two forms of flow control that can be established via auto-negotiation: symmetric and asymmetric. Symmetric flow control is for point-to-point links; asymmetric for hub-to-end-node connections. Symmetric flow control enables either node to flow-control the other. Asymmetric flow-control enables a repeater or switch to flow-control a DTE, but not vice versa.

Table 3-34 lists the intended operation for the various settings of *ASM_DIR* and *PAUSE*. This information is provided for reference only; it is the responsibility of the MAC to implement the correct function. The PHY merely enables the two MACs to communicate their abilities to each other.

Table 3-34. Pause And Asymmetric Pause Settings

ASM_DIR Settings Local (PHYREG 0,4.10) and Remote (PHYREG 0,5.10)	Pause Setting - Local (PHYREG 0,4.11)	Pause Setting - Remote (PHYREG 0,5.11)	Result
Both ASM_DIR = 1b	1b	1b	Symmetric - Either side can flow control the other.
	1b	0b	Asymmetric - Remote can flow control local only.
	0b	1b	Asymmetric - Local can flow control remote.
	0b	0b	No flow control.
Either or both ASM_DIR = 0b	1b	1b	Symmetric - Either side can flow control the other.
	Either or both = 0b		No flow control.



3.6.7.5 Management Data Interface

The PHY supports the IEEE 802.3 MII Management Interface also known as the Management Data Input/Output (MDIO) Interface. This interface enables upper-layer devices to monitor and control the state of the PHY. The MDIO interface consists of a specific protocol that runs across the connection, and an internal set of addressable registers.

The PHY supports the core 16-bit MDIO registers. Registers 0-10 and 15 are required and their functions are specified by the IEEE 802.3 specification. Additional registers are included for expanded functionality.

3.6.7.6 Internal PHY Low Power Operation and Power Management

The internal PHY incorporates numerous features to maintain the lowest power possible.

The PHY can be entered into a low-power state according to MAC control (Power Management controls) or via PHY Register 0. In either power down mode, the PHY is not capable of receiving or transmitting packets.

3.6.7.6.1 Power Down via the PHY Register

The PHY can be powered down using the control bit found in *PHYREG 0,0.11*. This bit powers down a significant portion of the port but clocks to the register section remain active. This enables the PHY management interface to remain active during register power down. The power down bit is active high. When the PHY exits software power-down (*PHYREG 0,0.11* = 0b), it re-initializes all analog functions, but retains its previous configuration settings.

3.6.7.6.2 Power Management State

The internal PHY tracks the device power states and device disable option. It provides the required Ethernet link functionality at an optimized power consumption. For additional details see the following sections within the "Power Management" chapter: "Entry to D0a State"; "Entry to D3 State"; "Entry to Dr State"; "Ultra Low Power - ULP" and "Internal PHY state vs. Device State".

3.6.7.6.3 Internal PHY Link Energy Detect

Foxville asserts the *Link Energy Detect* bit (*PHPM.Link Energy Detect*) each time energy is detected on the link. This bit provides an indication of a cable becoming plugged or unplugged.

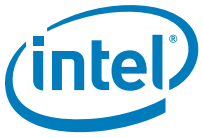
3.6.7.7 Advanced Diagnostics

Foxville integrated PHY incorporates hardware support for advanced diagnostics.

The hardware support enables output of internal PHY data to host memory for post processing by the software device driver.

The current diagnostics supported are described in the sections that follow.

3.6.7.7.1 Channel Frequency Response



By doing analysis on the Tx and Rx data, it can be established that a channel's frequency response (also known as insertion loss) can determine if the channel is within specification limits. (Clause 40.7.2.1 in IEEE 802.3).

4.0 Initialization

4.1 Power Up

4.1.1 Power-Up Sequence

Figure 4-1 shows the power-up sequence from power ramp up and to when Foxville is ready to accept host commands.

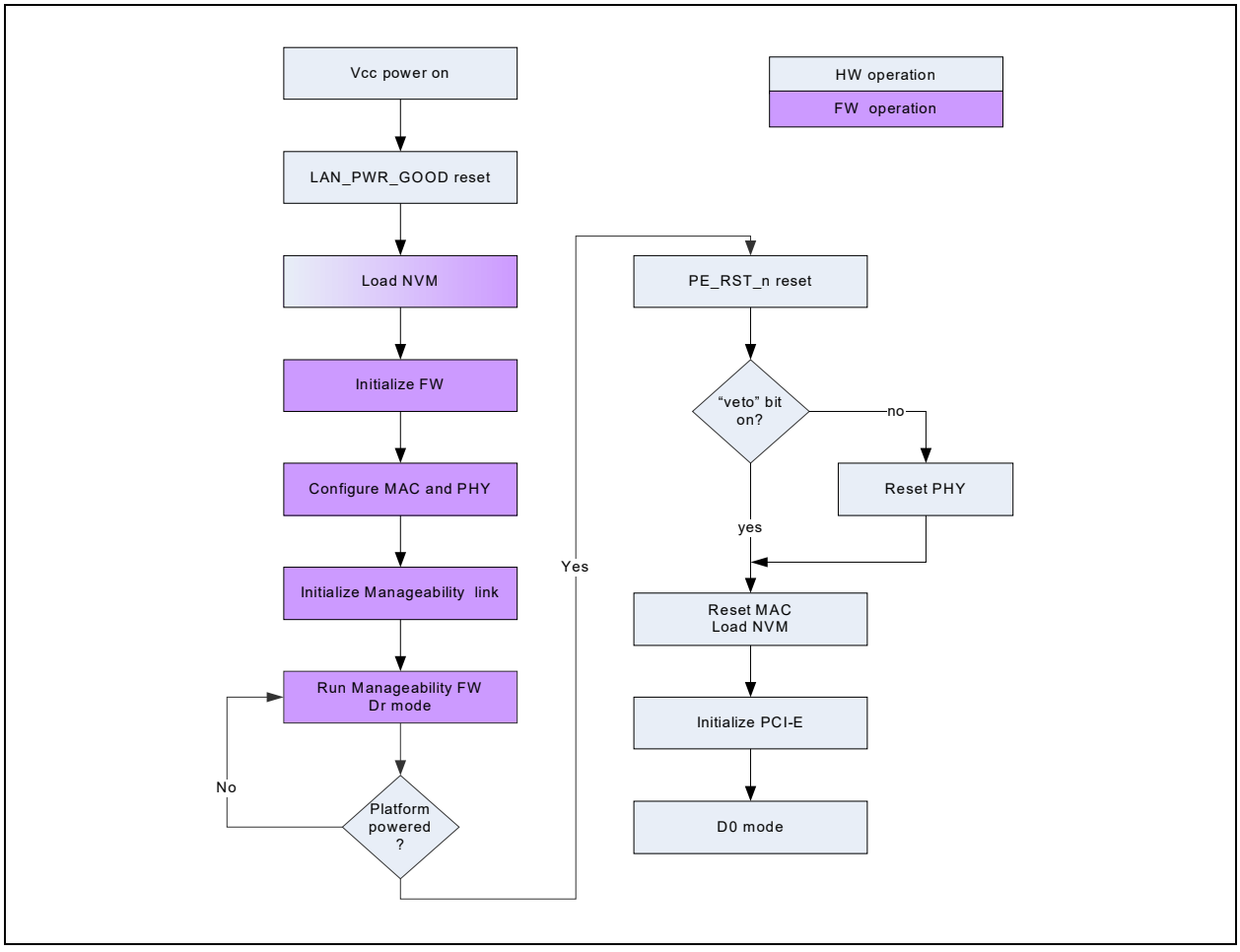
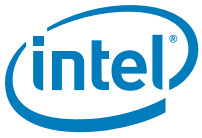


Figure 4-1. Power-Up - General Flow



Note: The Keep_PHY_Link_Up bit (*Veto* bit) is set by firmware when the MC is running IDER or SoL. Its purpose is to prevent interruption of these processes when power is being turned on.

4.2 Reset Operation

Foxville has a number of reset sources described in the following sections. After a reset, the software device driver should verify that the *EEMNGCTL.CFG_DONE* bit (refer to Section 8.4.7) is set to 1b and no errors were reported in the *FWSM.Ext_Err_Ind* (refer to Section 8.7.2) field.

4.2.1 Hardware Reset Sources

Foxville reset sources are described in the sections that follow.

4.2.1.1 Power On Reset

Foxville has an internal mechanism for sensing the power pins. Once power is up and stable and the external LAN_PWR_GOOD pin is high and internal clocks are stable, Foxville releases an internal power on reset. This reset acts as a master reset of the entire chip. It is level sensitive, and while it is zero holds all of the registers in reset.

4.2.1.2 PCIe Reset

De-asserting PE_RST_N indicates that both the power and the PCIe clock sources are stable. This pin asserts an internal reset also after a D3cold exit. Most units are reset on the rising edge of PE_RST_N. The only exception is the PCIe unit, which is kept in reset while PE_RST_N is asserted (level).

4.2.1.3 In-Band PCIe Reset

Foxville generates an internal reset in response to a physical layer message from the PCIe or when the PCIe link goes down (entry to polling or detect state). This reset is equivalent to PCIe reset.

4.2.1.4 D3hot to D0 Transition

This is also known as ACPI reset. Foxville generates an internal reset on the transition from D3hot power state to D0 (caused after configuration writes from D3 to D0 power state).

When the *PMCSR.No_Soft_Reset* bit in the configuration space is set, on transition from D3hot to D0 Foxville resets internal CSRs (similar to software reset *CTRL.DEV_RST*) but doesn't reset registers in the PCIe configuration space. If the *PMCSR.No_Soft_Reset* bit is cleared, Foxville resets all per-function registers except for registers defined as sticky in the configuration space.

Note: Regardless of the value of the *PMCSR.No_Soft_Reset* bit, the function is reset (including bits that are not defined as sticky in PCIe configuration space) if the link state has transitioned to the L2/L3 ready state, on transition from D3cold to D0, if Function Level Reset (FLR) is asserted or if transition D3hot to D0 is caused by asserting the PCIe reset (PE_RST pin).

Note: Software device drivers should implement the handshake mechanism defined in Section 5.2.3.3 to verify that all pending PCIe completions finish, before moving Foxville to D3.

4.2.2 OS Software Reset

4.2.2.1 FLR

A FLR function reset is issued by setting bit *15* in the *Device Control* configuration register, which is equivalent to a D0 ⇨ D3 ⇨ D0 transition. The only difference is that this reset does not require software device driver intervention in order to stop the master transactions of this function. The NVM content is partially reloaded after a FLR reset. The words read from NVM at FLR are the same as read following a full software reset. A list of these words can be found in Section 3.3.1.1.

A FLR reset to a function resets all the queues, interrupts, and statistics registers attached to the function. It also resets PCIe read/write configuration bits as well as disables transmit and receive flows for the queues allocated to the function. All pending read requests are dropped and PCIe read completions to the function might be completed as unexpected completions and silently discarded (following update of flow control credits) without logging or signaling as an error.

Note: If software initiates a FLR when the *Transactions Pending* bit in the *Device Status* configuration register is set to 1b (refer to Section 9.4.5.6), then software must not initialize the function until allowing time for any associated completions to arrive. The *Transactions Pending* bit is cleared upon completion of the FLR.

4.2.2.2 Bus Master Enable (BME)

Disabling bus master activity of a function by clearing the Configuration *Command register.BME* bit to 0b, resets all DMA activities and MSI/MSIx operations. The master disable resets only the DMA activities related to this function without affecting the MC functionality. Configuration accesses and target accesses to the function are still enabled.

Note: Prior to issuing a master disable the software device driver needs to implement the master disable algorithm as defined in Section 5.2.3.3. After *Master Enable* is set back to 1b,the software device driver should re-initialize the transmit and receive queues.

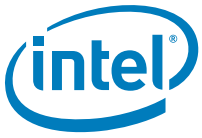
4.3 Software Driver Reset

4.3.1 Software Reset (DEV_RST)

Software can reset Foxville by setting the Software Reset (*CTRL.DEV_RST)* bit in the Device Control register. Following reset, the PCI configuration space (configuration and mapping) of the device is unaffected. Prior to issuing a software reset the software device driver needs to follow the master disable flow as defined in Section 5.2.3.3.

The *CTRL.DEV_RST* bit is provided primarily to recover from an indeterminate or suspected hung hardware state. Most registers (receive, transmit, interrupt, statistics, etc.) and state machines are set to their power-on reset values, approximating the state following a power-on or PCIe reset (refer to Table 4-2 for further information on affects of software reset). However, PCIe configuration registers and DMA logic is not reset, leaving the device mapped into system memory space and accessible by a software device driver.

To ensure that a software reset fully completed and that Foxville responds correctly to subsequent accesses after setting the *CTRL.DEV_RST* bit, the software device driver should wait at least 3 ms before accessing any register and then verify that *EEC.Auto_RD* is set to 1b and that the *STATUS.RST_DONE* bit is set to 1b.



When asserting the *CTRL.DEV_RST* software reset bit, only some NVM bits related to the specific function are re-read (refer to Section 3.3.1.1). Bits re-read from NVM are reset to default values.

4.3.2 NVM Reset

Writing a 1b to the NVM *Reset* bit of the Extended Device Control Register (*CTRL_EXT.EE_RST*) causes Foxville to re-read the function's configuration from the NVM. The *EE_RST bit is self cleared by the hardware.*

4.3.3 PHY Reset

The PHY is a shared resource between the host software and the MC (handled by the embedded firmware). Resetting the PHY, the host software is expected to coordinate it with the embedded firmware as follow. Reseting the PHY the PHPM register is expected to be used as well. As it is also a shared resource, software is expected to coordinate its access with the embedded firmware as well:

1. Check that the *Keep_PHY_Link_Up* flag *in the MANC* register is cleared. If set, the MC requires a stable link and thus the PHY should not be reset at this stage. The software device driver might skip the PHY reset if not mandatory or poll the *Keep_PHY_Link_Up* flag until it is cleared.
2. Take ownership of the PHY and the PHPM register by the following steps
   a. Take ownership of the software firmware synchronization register - *SW_FW_SYNC*:
      - Set the *SWSM.SWESMBI* bit.
      - Read *SWSM*.
      - Continue if the *SWSM.SWESMBI* is successfully set (indicting that the semaphore is acquired); otherwise, go back to step a.
   b. Software reads the Software-Firmware Synchronization Register (*SW_FW_SYNC)*.
      - If the FW_PHY_SM flag or the FW_MAC_CSR_SM flag are set (the firmware owns the PHY or the shared MAC CSRs) then release ownership of the software/firmware semaphore by clearing the *SWSM.SWESMBI* bit. Then either skip the PHY reset or wait about 100 µs and then go back to step 1.
      - Else (the FW_PHY_SM flag is cleared), then set the SW_PHY_SM flag indicating that the software owns the PHY interface and then release ownership of the software/firmware semaphore by clearing the *SWSM.SWESMBI* bit.
3. Read the PHPM register clearing the *RST_COMPL* flag (needed step prior to step #7).
4. Drive the PHY reset bit in the CTRL register (bit 31).
5. Wait 100 µs.
6. Release PHY reset bit in the CTRL register (bit 31).
7. Poll the *RST_COMPL* flag in the *PHPM* register until it is active (indicating the PHY reset completion). Note that the reset is supposed to be completed within approximate 100 µs.
8. Release ownership of the PHY and the MAC CSRs by the following steps:
   a. Take ownership of the software firmware synchronization register - *SW_FW_SYNC*:
      - Set the *SWSM.SWESMBI* bit.
      - Read *SWSM*.
      - Continue if the *SWSM.SWESMBI* is successfully set (indicting that the semaphore is acquired); otherwise, go back to step a.
   b. Clear the FW_PHY_SM and the FW_MAC_CSR_SM bits in *SW_FW_SYNC* that control the software ownership to indicate that these resources are free.

c. Release ownership of the software/firmware semaphore by clearing the *SWSM.SWESMBI* bit.

9. Wait until any required initialization of the PHY by the firmware is completed by polling the *CFG_DONE* flag in the *EEMNGCTL register until it is active.*
10. If there is a need for PHY configuration by the software then take PHY ownership, configure the PHY and then release the ownership.

4.4 Manageability Reset Interface

4.4.1 Force TCO

This reset is generated when manageability logic is enabled and the MC detects that Foxville does not receive or transmit data correctly. Force TCO reset is enabled if the *Reset on Force TCO* bit in the Management Control NVM word is set 1b. Table 4-2 describes affects of TCO reset on Foxville functionality.

Force TCO reset is generated in pass through mode when the MC issues a Force TCO command with bit 1 set and the previous conditions exist.

4.4.2 PHY Behavior During a Manageability Session

During some manageability sessions (such as an IDER or SoL session as initiated by an external MC), the platform is reset so that it boots from a remote media. This reset must not cause the Ethernet link to drop since the manageability session is lost. Also, the Ethernet link should be kept on continuously during the session for the same reasons. Foxville therefore limits the cases in which the internal PHY would restart the link, by masking two types of events from the internal PHY:

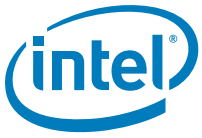
- PE_RST# and PCIe resets (in-band and link drop) do not reset the PHY during such a manageability session
- The PHY does not change link speed as a result of a change in power management state, to avoid link loss. For example, the transition to D3hot state is not propagated to the PHY.
  - Note however that if main power is removed, the PHY is allowed to react to the change in power state (the PHY might respond in link speed change). The motivation for this exception is to reduce power when operating on auxiliary power by reducing link speed.

The capability described in this section is disabled by default on LAN_POWER_GOOD reset. The *Keep_PHY_Link_Up_En* bit in the NVM must be set to 1b to enable it. Once enabled, the feature is enabled until the next LAN_POWER_GOOD (Foxville does not revert to the hardware default value on PE_RST#, PCIe reset or any other reset but LAN_POWER_GOOD).

When the *Keep_PHY_Link_Up* bit (also known as *Veto* bit) in the MANC register is set, the following behaviors are disabled:

- The PHY is not reset on PE_RST# and PCIe resets (in-band and link drop). Other reset events are not affected - LAN_POWER_GOOD 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.
- Foxville does not initiate configuration of the PHY to avoid losing link.

The *Keep_PHY_Link_Up* bit is set by the MC through the Management Control command on the sideband interface. It is cleared by the external MC (again, through a command on the sideband interface) when the manageability session ends. Once the *Keep_PHY_Link_Up* bit is cleared, the PHY updates its Dx state and acts accordingly (negotiates its speed).



The *Keep_PHY_Link_Up* bit is a read/write bit and can be accessed by host software, but software is not expected to clear the bit. The bit is cleared in the following cases:

- On LAN_POWER_GOOD.
- When the MC resets or initializes it.

4.5 Registers and Logic Reset Affects

The resets affect the following registers and logic:

Table 4-1. Foxville Reset Affects - Common Resets

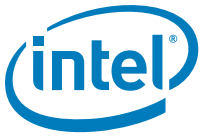
Reset Activation	LAN_PWR_GOOD	PE_ RST_N	In-Band PCIe Reset	FW Reset	Notes
LTSSM (PCIe Back to Detect/ Polling)	+	+	+		
PCIe Link Data Path	+	+	+		
Read NVM					16.
Read NVM (Complete Load)	+	+	+		
PCI Configuration Registers - Non Sticky	+	+	+		3.
PCI Configuration Registers - Sticky	+	+	+		4.
PCIe Local Registers	+	+	+		5.
Data Path	+	+	+		
On-die Memories	+	+	+		13.
MAC, PCS, Auto Negotiation and Other Port Related Logic	+	+	+		
DMA	+	+	+		
Functions Queue Enable	+	+	+		
Function interrupt and Statistics Registers	+	+	+		
Wake Up (PM) Context	+				7.
Wake Up Control Register	+				8.
Wake Up Status Registers	+				9.
Manageability Control Registers	+				10.
MMS Unit	+			+	
Wake-Up Management Registers	+	+	+		3.,11.
Memory Configuration Registers	+	+	+		3.
NVM Requests	+				14.
PHY	+	+	+		2.
Strapping Pins	+	+	+		

Table 4-2. Foxville Reset Affects - Other Resets

Reset Activation	D3hot -> D0	FLR	SW Reset (CTRL.DEV_RST)	Force TCO	EE Reset	PHY Reset	Notes
Port Configuration Auto-load from NVM	+	+	+	+	+		
PCI Configuration Registers Read Only							3.
PCI Configuration Registers MSI-X	+	+					6.
PCI Configuration Registers Read/Write							
PCIe Local Registers							5.
Data Path	+	+	+	+			
On-die Memories	+	+	+	+			13.
MAC, PCS, Auto Negotiation and Other Port Related Logic	+	+	+	+			
DMA	+	+		+			15.
Wake Up (PM) Context							7.
Wake Up Control Register							8.
Wake Up Status Registers							9.
Manageability Control Registers							10.
Function Queue Enable	+	+	+	+			
Function Interrupt and Statistics Registers	+	+	+				
Wake-Up Management Registers	+	+	+	+			3.,11.
Memory Configuration Registers	+	+	+	+			3.
Flash Request	+	+					14.
PHY	+	+		+		+	2.
Strapping Pins							

Notes:

1. If AUX_POWER = 0b the Wakeup Context is reset (*PME_Status* and *PME_En* bits should be 0b at reset if Foxville does not support PME from D3cold).
2. The MMS unit must configure the PHY after any PHY reset.
3. The following register fields do not follow the general rules previously described:
   a. *CTRL.SDP0_IODIR, CTRL.SDP1_IODIR, CTRL_EXT.SDP2_IODIR, CTRL_EXT.SDP3_IODIR, CONNSW.ENRGSRC field, CTRL_EXT.EXT_VLAN* and LED configuration registers are reset on LAN_PWR_GOOD only. Any NVM read resets these fields to the values in the NVM.
   b. The *Aux Power Detected* bit in the PCIe Device Status register is reset on LAN_PWR_GOOD and PE_RST_N (PCIe reset) assertion only.
   c. FLA - reset on LAN_PWR_GOOD only.
   d. The bits mentioned in the next note.
4. The following registers are part of this group:
   a. VPD registers



b. Max payload size field in PCIe Capability Control register (offset 0xA8).
c. *Active State Link PM Control* field, *Common Clock Configuration* field and *Extended Synch* field in PCIe Capability Link Control register (Offset 0xB0).
d. Read *Completion Boundary* in the PCIe Link Control register (Offset 0xB0).

5. The following registers are part of this group:
   a. SWSM
   b. *GCR* (only part of the bits - see register description for details)
   c. FUNCTAG
   d. *SW_FW_SYNC* - only part of the bits - see register description for details.
6. The following registers are part of this group:
   a. *MSIX control* register, *MSIX PBA* and *MSIX per vector mask*.
7. The *Wake Up Context* is defined in the PCI Bus Power Management Interface Specification (sticky bits). It includes:
   — *PME_En* bit of the Power Management Control/Status Register (*PMCSR*).
   — *PME_Status* bit of the Power Management Control/Status Register (*PMCSR*).
   — *Aux_En* in the PCIe registers
   — The device Requester ID (since it is required for the PM_PME TLP).
   The shadow copies of these bits in the Wakeup Control register are treated identically.
8. Refers to bits in the Wake Up Control register that are not part of the Wake-Up Context (the *PME_En* and *PME_Status* bits).
9. The Wake Up Status registers include the following:
   a. Wake Up Status register
   b. Wake Up Packet Length.
   c. Wake Up Packet Memory.
10. The Manageability Control registers refer to the following registers:
    a. *MANC* 0x5820
    b. FWSM
11. The Wake-up Management registers include the following:
    a. Wake Up Filter Control
    b. IP Address Valid
    c. IPv4 Address Table
    d. IPv6 Address Table
    e. Flexible Filter Length Table
    f. Flexible Filter Mask Table
12. The other configuration registers include:
    a. General Registers
    b. Interrupt Registers
    c. Receive Registers
    d. Transmit Registers
    e. Statistics Registers
    f. Diagnostic Registers

The registers: *MTA[n], VFTA[n], WUPM[n], FHFT[n], FHFT_EXT[n], TDBAH/TDBAL*, and *RDBAH/RDVAL* have no default value. If the functions associated with the registers are enabled, they must be programmed by software. Once programmed, their value is preserved through all resets as long as power is applied to Foxville.

Note: In situations where the device is reset using the software reset *CTRL.DEV_RST,* the transmit data lines are forced to all zeros. This causes a substantial number of symbol errors detected by the link partner.

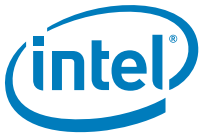
13. The contents of the following memories are cleared to support the requirements of PCIe FLR:
    a. The Tx packet buffers
    b. The Rx packet buffers
14. Includes *EEC.REQ*, *EEC.GNT*, *FLA.REQ* and *FLA.GNT* fields.
15. The following DMA registers are cleared only by LAN_PWR_GOOD, PCIe Reset or *CTRL.DEV_RST*: *DTPARS, DRPARS* and *DDPARS.*
16. *CTRL.DEV_RST* assertion causes read of function related sections.

4.5.1 Registers Initialization by Software

This section lists registers that should be initialized by software to a different values than its internal default values.

Table 4-3. Foxville Reset Affects - Common Resets

Register	Address	Bit / Field	Default Value	Required Software Setting	Comment
TCTL	0x0400	BST	0x40	0x3F	Required setting for half duplex
TQAVCTRL	0x3570	bit #3	0b	1b	SW should set this bit to '1' when TSN is enabled
RETA	0x5C00 + 4*n	The whole register array	X	0x0	SW should initialize the table before enabling multiple queues
EEE_SU	0x0E34	LPI_Clock_Stop	1b	0b	SW should clear this bit to '0' as part of the driver init flow
TIMADJ_0	0xB60C	bit #30	0b	1b	Required setting when the SW sets the TIMADJ_0 register
PCIe_DIG_Delay	0x12550	DIG1_TX_LATE	0x2B	0x00	Required initialization if PCIe PTM functionality is used
		DIG2_TX_LATE	0x17	0x00	
		DIG1_RX_LATE	0x4B	0x44	
		DIG2_RX_LATE	0x27	0x01	
PCIe_PHY_DELAY	0x12554	PHY1_TX_LATE	0x2A	0x00	Required initialization if PCIe PTM functionality is used
		PHY2_TX_LATE	0x15	0x00	
		PHY1_RX_LATE	0x80	0x90	
		PHY2_RX_LATE	0x40	0x64	



4.6 Device and Function Disable

4.6.1 General

For a LAN on Motherboard (LOM) design, it might be desirable for the system to provide BIOS-setup capability for selectively enabling or disabling LAN functions. It enables the end-user more control over system resource-management and avoid conflicts with add-in NIC solutions. Foxville provides support for selectively enabling or disabling the device.

Device presence (or non-presence) must be established early during BIOS execution, in order to ensure that BIOS resource-allocation (of interrupts, of memory or I/O regions) is done according to devices that are present only. This is frequently accomplished using a BIOS Configuration Values Driven on Reset (CVDR) mechanism. Foxville LAN-disable mechanism is implemented in order to be compatible with such a solution.

4.6.2 Disabling Both LAN Port and PCIe Function (Device Off)

Foxville provides a mechanism to disable its LAN port and the PCIe function. When LAN_DISABLE_N is asserted (driven low).

While in device disable mode, the PCIe link is in L3 state. The PHY is in power down mode. Output buffers are tri-stated.

Asserting or deasserting PCIe PE_RST_N does not have any affect while the device is in device disable mode (the device stays in the respective mode as long as the right settings on LAN_DISABLE_N). However, the device might momentarily exit the device disable mode from the time PCIe PE_RST_N is de-asserted again and until the NVM is read.

Driving the LAN_DISABLE_N signal (to low), Foxville can also enter the Ultra Low Power (ULP) state if enabled in the NVM. See additional description in Section 5.6.1.1.

During power-up, the input pin LAN_DISABLE_N is ignored until the NVM is read. From that point, the device might enter device disable according to the NVM settings.

4.6.3 Disabling PCIe Function Only

Foxville also supports disabling just the PCIe function but keeping the LAN port that resides on it fully active (for manageability purposes and BMC pass-through traffic). This functionality can be achieved by setting NVM en_pin_pcie_func_dis bit (word 0x29) to 1b and driving the MCS / PCIE_ENA pin to low during power up or exit from the ULP state. There are two possible use cases for this option:

1. Foxville is designed by the system manufacturer as a LAN connection ONLY for the CSME. In this case, the en_pin_pcie_func_dis bit in the NVM is pre-programmed to 1b and the MCS / PCIE_ENA pin should be connected to a 10K ohm pull down resistor.
2. Foxville is designed by the system manufacturer as a nominal LAN connection with an option for the end user to enable it ONLY for the CSME. In this case, the en_pin_pcie_func_dis bit in the NVM is pre-programmed to 1b and the MCS / PCIE_ENA pin should be connected by a 10K ohm resistor to a logic output controlled possibly by a BIOS setting option. In this case the following flow is needed:
   - The en_pin_pcie_func_dis bit in the NVM should be pre-programmed to 1b.
   - Drive the LAN_DISABLE_N signal to low transiting Foxville to the ULP state.
   - Drive the MCS / PCIE_ENA pin through a 10K ohm resistor to a logic low level.

— De-assert the LAN_DISABLE_N signal to the high level transiting Foxville back to active state. During this transition Foxville samples the MCS / PCIE_ENA strapping and the PCIe function is disabled. Note that the LAN_DISABLE_N signal should be kept at low for at least 0.1 sec.

4.6.4 BIOS Handling of Device Disable

See Section 5.6, Section 5.6.1 and Section 5.6.1.1 for description.

4.7 Software Initialization and Diagnostics

4.7.1 Introduction

This section discusses general software notes for Foxville, especially initialization steps. This includes general hardware, power-up state, basic device configuration, initialization of transmit and receive operation, link configuration, software reset capability, statistics, and diagnostic hints.

4.7.2 Power Up State

When Foxville powers up it reads the NVM. The NVM contains sufficient information to bring the link up and configure Foxville for manageability and/or APM wakeup. However, software initialization is required for normal operation.

The power-up sequence, as well as transitions between power states, are described in Section 4.1.1. The next section gives more details on configuration requirements.

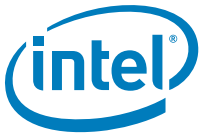
4.7.3 Initialization Sequence

The following sequence of commands is typically issued to the device by the software device driver in order to initialize Foxville to normal operation. The major initialization steps are:

- Disable Interrupts - see Interrupts during initialization.
- Issue Software Reset and perform General Configuration - see Section 4.7.5.
- Setup the PHY and the link - see Section 4.7.7.
- Initialize all statistical counters - see Section 4.7.8.
- Initialize Receive - see Section 4.7.9.
- Initialize Transmit - see Section 4.7.10.
- Enable Interrupts - see Section 4.7.4.

4.7.4 Interrupts During Initialization

- Most drivers disable interrupts during initialization to prevent re-entering the interrupt routine. Interrupts are disabled by writing to the Extended Interrupt Mask Clear (EIMC) register. Note that the interrupts need to be disabled also after issuing a software reset, so a typical driver initialization flow is:
- Disable interrupts
- Issue a Software Reset
- Disable interrupts (again)
- …



After initialization completes, a typical software device driver enables the desired interrupts by writing to the Extended Interrupt Mask Set (EIMS) register.

4.7.5 Software Reset and General Configuration

Device initialization typically starts with a software reset that places the device into a known state and enables the software device driver to continue the initialization sequence. If the device state is unknown to the software driver, it is recommended to initiate a device reset by setting the CTRL.DEV_RST. Note that prior to issuing the reset, the software needs to follow the master disable flow as defined in Section 5.2.3.3.

Several values in the Device Control (CTRL) register need to be set, upon power up, or after a device reset for normal operation.

- The *FD* bit should be set per interface negotiation (if done in software), or is set by the hardware if the interface is auto-negotiating. This is reflected in the Device Status Register in the auto-negotiation case.
- Speed is determined via auto-negotiation by the PHY, or forced by software if the link is forced. Status information for speed is also readable in the STATUS register.
- The *ILOS* bit should normally be set to 0b.

4.7.6 Flow Control Setup

If flow control is enabled, program the FCRTL0, FCRTH0, FCTTV and FCRTV registers. In order to avoid packet losses, FCRTH should be set to a value equal to at least two maximum size packets below the receive buffer size (assuming a packet buffer size of 36 KB and the expected maximum size packet of 9.5 KB), the FCRTH0 value should be set to 36 - 2 * 9.5 = 17KB.For example, FCRTH0.RTH should be set to 0x440.

The receive buffer size is controlled by RXPBSIZE.Rxpbsize register field. Refer to Section 4.7.9 for its setting rules.

4.7.7 Link Setup Mechanisms and Control/Status Bit Summary

4.7.7.1 PHY Initialization

Refer to the PHY documentation for the initialization and link setup steps. The software device driver uses the MDIC register to initialize the PHY and setup the link. Section 3.6.3.1 describes the link setup for the internal copper PHY. Section 3.6.2.2 Section describes the usage of the MDIC register.

4.7.7.2 MAC/PHY Link Setup

This section summarizes the proper MAC/PHY link setups.

4.7.7.2.1 MAC Settings Automatically Based on Duplex and Speed Resolved by PHY (CTRL.FRCDPLX = 0b, CTRL.FRCSPD = 0b,)

CTRL.FD	Don't care; duplex setting is established from PHY's internal indication to the MAC (FDX) after PHY has auto-negotiated a successful link-up.
CTRL.SLU	Must be set to 1b by software to enable communications between MAC and PHY.

CTRL.RFCE	Must be programmed by software after reading capabilities from PHY registers and resolving the desired flow control setting.
CTRL.TFCE	Must be programmed by software after reading capabilities from PHY registers and resolving the desired flow control setting.
CTRL.SPEED	Don't care; speed setting is established from PHY's internal indication to the MAC (*SPD_IND*) after PHY has auto-negotiated a successful link-up.
STATUS.FD	Reflects the actual duplex setting (FDX) negotiated by the PHY and indicated to MAC.
STATUS.LU	Reflects link indication (LINK) from PHY qualified with *CTRL.SLU* (set to 1b).
STATUS.SPEED	Reflects actual speed setting negotiated by the PHY and indicated to the MAC (*SPD_IND*).

4.7.8 Initialization of Statistics

Statistics registers are hardware-initialized to values as detailed in each particular register's description. The initialization of these registers begins upon transition to D0active power state (when internal registers become accessible, as enabled by setting the *Memory Access Enable* bit of the PCIe Command register), and is guaranteed to be completed within 1 μs of this transition. Access to statistics registers prior to this interval might return indeterminate values.

All of the statistical counters are cleared on read and a typical device driver reads them (thus making them zero) as a part of the initialization sequence.

4.7.9 Receive Initialization

Program the receive address register(s) per the station address. This can come from the NVM or by any other means (for example, on some machines, this comes from the system PROM not the NVM on the adapter card).

Set up the Multicast Table Array (MTA) by software. This means zeroing all entries initially and adding in entries as requested.

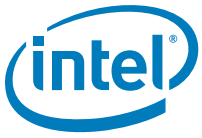
Program the RXPBSIZE register so that the total size formed by the receive packet buffer plus the BMC to OS buffer plus the transmit packet buffer(s) plus the OS to BMC buffer does not exceed 64 KB:

```
RXPBSIZE.Rxpbsize + RXPBSIZE.Bmc2ospbsize + TXPBSIZE.Txpb0size + TXPBSIZE.Txpb1size +
TXPBSIZE.Txpb2size + TXPBSIZE.Txpb3size + TXPBSIZE.os2Bmcpbsize <= 64 KB
```

Program *RCTL* with appropriate values. It is best to leave the receive logic disabled (*RCTL.RXEN* = 0b) until after the receive descriptor rings have been initialized. If VLANs are not used, software should clear *VFE*. Then there is no need to initialize the *VFTA*. Select the receive descriptor type.

The following should be done once per receive queue that is planned to be used:

1. Allocate a region of memory for the receive descriptor list.
2. Receive buffers of appropriate size should be allocated and pointers to these buffers should be stored in the descriptor ring.
3. Program the descriptor base address with the address of the region.
4. Set the length register to the size of the descriptor ring.
5. Program *SRRCTL* of the queue according to the size of the buffers, the required header handling and the drop policy.



6. If header split or header replication is required for this queue, program the *PSRTYPE* register according to the required headers.

- Program the RXDCTL.*WTHRESH* with the desired Rx descriptor write back policy. Suggested value for good PCIe BW utilization is around 0x4. All other fields can be kept at their default values.

7. Enable the queue by setting *RXDCTL.ENABLE*. In the case of queue zero, the enable bit is set by default - so the ring parameters should be set before *RCTL.RXEN* is set.
8. Poll the *RXDCTL* register until the *ENABLE* bit is set.
9. Program the direction of packets to this queue according to the mode selected in the *MRQC* register. Packets directed to a disabled queue are dropped.
10. Bump the queue tail register pointing to the last descriptor of the queue.

4.7.9.1 Initialize the Receive Control Register

To properly receive packets the receiver should be enabled by setting *RCTL.RXEN*. This should be done only after all other setup is accomplished. If software uses the Receive Descriptor Minimum Threshold Interrupt, that value should be set.

4.7.9.2 Dynamic Enabling and Disabling of Receive Queues

Receive queues can be dynamically enabled or disabled given the following procedure is followed:

Enabling a queue:

- Follow the per queue initialization sequence described in Section 4.7.9.

Note: If there are still packets in the packet buffer assigned to this queue according to previous settings, they are received after the queue is re-enabled. In order to avoid this condition, the software might poll the *PBWAC* register. Once a an empty condition of the relevant packet buffer is detected or two wrap around occurrences are detected the queue can be re-enabled.

Disabling a Queue:

1. Disable the packet assignments to this queue.
2. Poll the PBWAC register until an empty condition of the relevant packet buffer is detected or two wrap around occurrences are detected.
3. Disable the queue by clearing *RXDCTL.ENABLE*. Foxville stops fetching and writing back descriptors from this queue immediately. Foxville eventually completes the storage of one buffer allocated to this queue. Any further packet directed to this queue is dropped. If the currently processed packet is spread over more than one buffer, all subsequent buffers are not written.
4. Foxville clears *RXDCTL.ENABLE* only after all pending memory accesses to the descriptor ring or to the buffers are done. The software device driver should poll this bit before releasing the memory allocated to this queue.

Note: The Rx path can be disabled only after all Rx queues are disabled.

4.7.10 Transmit Initialization

- Program the TCTL register according to the MAC behavior needed.
- Program the TXPBSIZE register so any transmit buffer that is in use is at least greater to twice the maximum packet size that might be stored in it. In addition, comply to the setting rules defined in Section 4.7.9.

The following should be done once per transmit queue:

- Allocate a region of memory for the transmit descriptor list.
- Program the descriptor base address with the address of the region.
- Set the length register to the size of the descriptor ring.
- Program the TXDCTL register with the desired Tx descriptor write back policy. Suggested values are:
  - *WTHRESH* = 0x1 for minimum latency or 0x4 for better PCIe BW utilization
  - All other fields 0b.
- If needed, set *TDWBAL*/*TWDBAH* to enable head write back.
- Enable the queue using *TXDCTL.ENABLE* (queue zero is enabled by default).
- Poll the TXDCTL register until the *ENABLE* bit is set.

Enable transmit path by setting *TCTL.EN*. This should be done only after all other settings are done.

At this point software can add transmit descriptors to the queue and bump the queue tail, indicating these descriptors to the device.

4.7.10.1 Dynamic Queue Enabling and Disabling

Transmit queues can be dynamically enabled or disabled given the following procedure is followed:

Enabling:

- Follow the per queue initialization described in the previous section.

Disabling:

- Stop storing packets for transmission in this queue.
- Wait until the head of the queue (*TDH)* is equal to the tail (*TDT*); the queue is empty.
- Disable the queue by clearing *TXDCTL.ENABLE*.

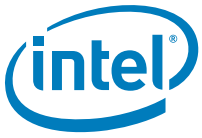
The Tx path might be disabled only after all Tx queues are disabled.

4.7.11 Alternate MAC Address Support

In some systems, the MAC address needs to be replaced with a temporary MAC address in a way that is transparent to the software layer. One possible usage is in blade systems, to enable a standby blade to use the MAC address of another blade that failed, so that the network image of the entire blade system does not change.

In order to enable this mode, a management console might change the MAC address in the NVM image. It is important in this case to be able to keep the original MAC address of the device as programmed at the factory.

In order to support this mode, Foxville provides the *Alternate Ethernet MAC Address* NVM structure to store the original MAC address. This structure is described in the "Alternate MAC Address Location" word in the NVM. When the MAC address is changed, the factory MAC address should be written to the *Alternate Ethernet MAC Address* structure before writing the new Ethernet MAC address to the ports *Ethernet Address* NVM words (refer to Section 6.1.1.1, Section 6.1.1.2, Section 6.1.1.3).



In some systems, it might be advantageous to restore the original MAC address at power on reset, to avoid conflicts where two network controllers would have the same MAC address. At power up, Foxville restores the LAN MAC addresses stored in the *Alternate Ethernet MAC Address* NVM structure to the regular *Ethernet MAC address* NVM words (refer to Section 6.1.1.1, Section 6.1.1.2, Section 6.1.1.3) if the following conditions are met:

1. The *restore MAC address* bit in the *Common Firmware Parameters* NVM word is set (Section 6.1.2.4).
2. The value in word 0x37 in the "Alternate MAC Address Location" word in the NVM is not 0xFFFF.
3. The MAC address set in the regular *Ethernet MAC Address* NVM words is different than the address stored in the *Alternate Ethernet MAC* address NVM structure.
4. The address stored in the alternate Ethernet MAC address structure is valid (not all zeros or all ones).

If the factory MAC address was restored by the internal firmware, the *FWSM.Factory MAC address restored* bit is set.

If the value at word 0x37 is valid, but the MAC addresses in the alternate MAC structure are not valid (0xFFFFFFFF), the regular MAC address is backed up in the alternate MAC structure.

Notes: If the MAC address is modified by Firmware, the RAH/RAL[0] registers should also be programmed by Firmware to contain the new address so that APM wake up operates with the correct address.

Whenever the *Ethernet MAC Address* NVM words are restored, firmware shall recompute and update the NVM checksum word 0x3F at the same occasion.

4.8 Access to Shared Resources

Part of the resources in Foxville are shared between several software entities - namely the driver and the internal firmware. In order to avoid contentions, a software device driver that needs to access one of these resources should use the flow described in Section 4.8.1 in order to acquire ownership of this resource and use the flow described in Section 4.8.2 in order to relinquish ownership of this resource.

The shared resources are:

1. NVM.
2. PHYs port.
3. CSRs accessed by the internal firmware after the initialization process. Currently there are no such CSRs.
4. SVR/LVR control registers.
5. Management Host Interface
6. I^2C register set

Note: Any other software tool that accesses the register set directly should also follow the flow described in the sections that follow.

4.8.1 Acquiring Ownership Over a Shared Resource

The following flow should be used to acquire a shared resource:

1. Get ownership of the software/software semaphore SWSM.SMBI (offset 0x5B50 bit 0).
   a. Read the SWSM register.
   b. If SWSM.SMBI is read as zero, the semaphore was taken.

c. Otherwise, go back to step a.

This step assures that other software will not access the shared resources register (SW_FW_SYNC).

2. Get ownership of the software/firmware semaphore SWSM.SWESMBI (offset 0x5B50 bit 1):
   a. Set the SWSM.SWESMBI bit.
   b. Read SWSM.
   c. If SWSM.SWESMBI was successfully set - the semaphore was acquired - otherwise, go back to step a.

   This step assures that the internal firmware will not access the shared resources register (SW_FW_SYNC).
3. Software reads the Software-Firmware Synchronization Register (SW_FW_SYNC) and checks both bits in the pair of bits that control the resource it wants to own.
   a. If both bits are cleared (both firmware and other software does not own the resource), software sets the software bit in the pair of bits that control the resource it wants to own.
   b. If one of the bits is set (firmware or other software owns the resource), software tries again later.
4. Release ownership of the software/software semaphore and the software/firmware semaphore by clearing SWSM.SMBI and SWSM.SWESMBI bits.
5. At this stage, the shared resources is owned by the software device driver and it might access it. The SWSM and SW_FW_SYNC registers can now be used to take ownership of another shared resources.

Note: Software ownership of SWSM.SWESMBI bit should not exceed 100 ms. If software takes ownership for a longer duration, firmware might implement a timeout mechanism and take ownership of the SWSM.SWESMBI bit.

Note: Software ownership of bits in the SW_FW_SYNC register should not exceed 1 second. If software takes ownership for a longer duration, firmware might implement a timeout mechanism and take ownership of the relevant SW_FW_SYNC bits.

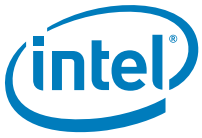
4.8.2 Releasing Ownership Over a Shared Resource

The following flow should be used to release a shared resource:

1. Get ownership of the software/software semaphore SWSM.SMBI (offset 0x5B50 bit 0).
   a. Read the SWSM register.
   b. If SWSM.SMBI is read as zero, the semaphore was taken.
   c. Otherwise, go back to step a.

   This step assures that other software will not access the shared resources register (SW_FW_SYNC).
2. Get ownership of the software/firmware semaphore SWSM.SWESMBI (offset 0x5B50 bit 1):
   a. Set the SWSM.SWESMBI bit.
   b. Read SWSM.
   c. If SWSM.SWESMBI was successfully set - the semaphore was acquired - otherwise, go back to step a.

   This step assures that the internal firmware will not access the shared resources register (SW_FW_SYNC).



3. Clear the bit in SW_FW_SYNC that controls the software ownership of the resource to indicate this resource is free.
4. Release ownership of the software/software semaphore and the software/firmware semaphore by clearing SWSM.SMBI and SWSM.SWESMBI bits.
5. At this stage, the shared resource are released by the driver and it may not access it. The SWSM and SW_FW_SYNC registers can now be used to take ownership of another shared resource.

5.0 Power Management

This section describes how power management is implemented in Foxville. Foxville supports the Advanced Configuration and Power Interface (ACPI) specification as well as Advanced Power Management (APM).

5.1 General Power State Information

5.1.1 PCI Device Power States

The PCIe Specification defines function power states (D-states) that enable the platform to establish and control power states for Foxville ranging from fully on to fully off (drawing no power) and various in-between levels of power-saving states, annotated as D0-D3. Similarly, PCIe defines a series of link power states (L-states) that work specifically within the link layer between Foxville and its upstream PCIe port (typically in the host chipset).

For a given device D-state, only certain L-states are possible as follows.

- D0 (fully on): Foxville is completely active and responsive during this D-state. The link can be in either L0 or deeper link power state, L1.
- D1 and D2: These modes are not supported by Foxville.
- D3 (off): Two sub-states of D3 are supported:
  - — D3hot, where primary power is maintained.
  - — D3cold, where primary power is removed.

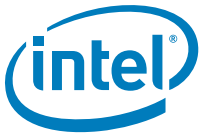
  Link states are mapped into device states as follows:
  - — D3hot maps to L1 to support clock removal on mobile platforms
  - — D3cold maps to L2 if auxiliary power is supported on Foxville with wake-capable logic, or to L3 if no power is delivered to Foxville. A sideband PE_WAKE_N mechanism is supported to interface wake-enabled logic on mobile platforms during the L2 state.

5.1.2 PCIe Link Power States

Table 5-1 lists allowable mapping from D-states to L-states on the PCIe link.

Table 5-1. Mapping From D-States to L-States

Downstream Component D-State[1]	Permissible Upstream Component D-State	Permissible L-State
D0	D0	L0, or L1
D3hot	D0-D3hot	L1
D3cold	D0-D3cold	L2 or L3



1. Refers to the maximum of Foxville state between all the functions sharing the link. For example, if function 0 is in D0 and function 1 is in D3, Foxville is considered to be in D0.

Configuring Foxville into a D-state automatically causes the PCIe link to transition to the appropriate L-state.

- L2/L3 Ready: This link state prepares the PCIe link for the removal of power and clock. Foxville is in the D3hot state and is preparing to enter D3cold. The power-saving opportunities for this state include, but are not limited to, clock gating of all PCIe architecture logic, shutdown of the PLL, and shutdown of all transceiver circuitry.
- L2: This link state is intended to comprehend D3cold with auxiliary power support. Note that sideband PE_WAKE_N signaling exists to cause wake-capable devices to exit this state. The power-saving opportunities for this state include, but are not limited to, shutdown of all transceiver circuitry except detection circuitry to support exit, clock gating of all PCIe logic, and shutdown of the PLL as well as appropriate platform voltage and clock generators.
- L3 (link off): Power and clock are removed in this link state, and there is no auxiliary power available. To bring Foxville and its link back up, the platform must go through a boot sequence where power, clock, and reset are reapplied appropriately.

5.2 Internal Power States

Foxville supports the D0 and D3 architectural power states as described earlier. Internally, Foxville 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 or a PCIe in-band reset is received

Figure 5-1 shows the power states and transitions between them.

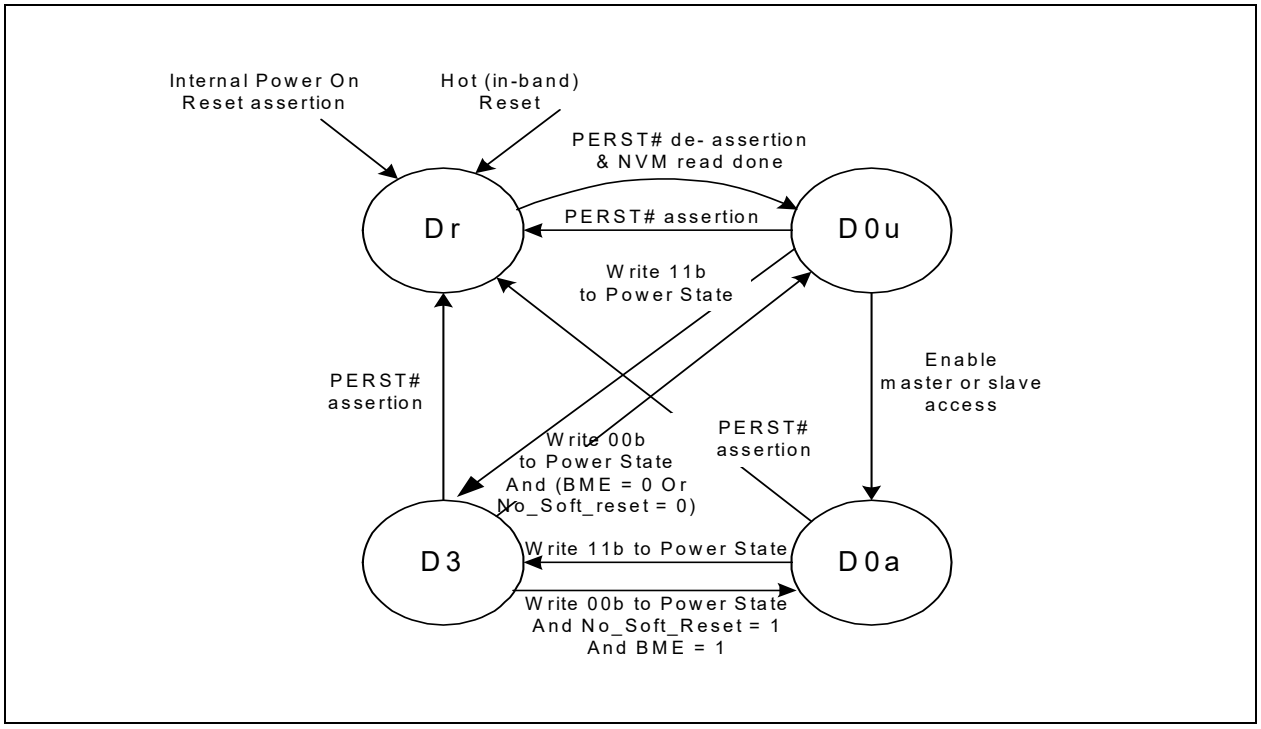


Figure 5-1. Power Management State Diagram

5.2.1 D0 Uninitialized State (D0u)

The D0u state is an architectural low-power state.

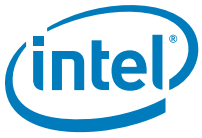
When entering D0u, Foxville:

- Disables wake up. Note that APM wake up is enabled (See additional information in Section 5.5.1), only if all of the following conditions are met:
  - The *WUC.APME* bit is set to 1b.
  - The *WUC.APMPME* bit or the *PMCSR.PME_en* bits are set to 1b.
  - The *WUC.EN_APM_D0* bit is set to 1b.

5.2.1.1 Entry into D0u state

D0u is reached from either the Dr state (on de-assertion of *PE_RST_N*) or the D3hot state (by configuration software writing a value of 00b to the *Power State* field of the PCI PM registers).

De-asserting *PE_RST_N* means that the entire state of Foxville is cleared, other than sticky bits. State is loaded from the NVM, followed by establishment of the PCIe link. Once this is done, configuration software can access Foxville.



On a transition from D3hot state to D0u state, Foxville PCI configuration space is not reset (since the *No_Soft_Reset* bit in the *PMCSR* register is set to 1b).

5.2.2 D0active State (D0a)

Once memory space is enabled, Foxville enters the D0 active 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.

Notes:

1. If the WUC.*EN_APM_D0* is cleared to 0b an APM wake event due to reception of a Magic packet is not generated when the function is not in D3 (or Dr) state. Any APM wake up previously active remains active when moving from D3 to D0.
2. If APM wake is required in D3 software device driver should not disable APM wake-up via the *WUC.APME* bit on D0 entry. Otherwise APM wake following a system crash and entry into S3, S4 or S5 system power management state is not enabled.

5.2.2.1 Entry to D0a State

D0a is entered from the D0u state by writing a 1b to the *Memory Access Enable* or the *I/O Access Enable* bit of the PCI Command register (See Section 9.3.3). The DMA, MAC, and PHY of the appropriate LAN function are also enabled. In order to reduce power consumption, the network link might have been previously in a low link speed. Entering the D0a state, the PHY can auto-negotiate to a higher link speed according to the device settings.

- If the PHY was at power down state then it exits this state.
- Else, if the link was disconnected then the link is set on.

Notes:

1. Following entry into D0a, the software device driver is expected to perform a full re-initialization of the function
2. Following entry into D0a, the software device driver can activate other wake-up filters by writing to the Wake Up Filter Control (*WUFC*) register (if instructed by the OS).

5.2.3 D3 State (PCI-PM D3hot)

Foxville transitions 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. If the *PMCSR.No_Soft_reset* bit is cleared upon completion or during the transition to D3 state, Foxville clears the *Memory Access Enable* and *I/O Access Enable* bits of the PCI Command register, which disables memory access decode. If the *PMCSR.No_Soft_reset* bit is set Foxville doesn't clear any bit in the PCIe configuration space. While in D3, Foxville 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.

5.2.3.1 Entry to D3 State

As mentioned above, transition to 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 Foxville; it masks all interrupts and does not write to the Transmit Descriptor Tail (TDT) register or to the Receive Descriptor Tail (RDT) register and operates the master disable algorithm as defined in Section 5.2.3.3.

As a response to being programmed into D3 state, Foxville transitions its PCIe link into the L1 link state. As part of the transition into L1 state, Foxville suspends scheduling of new TLPs and waits for the completion of all previous TLPs it has sent. If the *PMCSR.No_Soft_reset* bit is cleared, Foxville clears the *Memory Access Enable* and *I/O Access Enable* bits of the PCI Command register, which disables memory access decode. Any receive packets that have not been transferred into system memory are kept in Foxville (and discarded later on D3 exit). Any transmit packets that have not been sent can still be transmitted (assuming the Ethernet link is up).

If wake up capability is needed, the system should enable wake capability by setting to 1b the *PME_En* bit in the PMCSR PCIe configuration register. After wake capability has been enabled, the software device driver should set up the appropriate wake up registers (*WUC*, *WUFC* and associated filters) prior to the D3 transition.

If Protocol offload (Proxying) capability is required and the *MANC.MPROXYE* bit is set to 1b, the software device driver should:

1. Send to the firmware the relevant protocol offload information (type of protocol offloads required, MAC and IPv4/6 addresses information for protocol offload) via the shared RAM Firmware/Software Host interface as defined in Section 8.22.1, Section 10.6 and Section 10.6.2.
2. Program the *PROXYFC* register and associated filters according to the protocol offload required.
3. Program t*he WUC.PPROXYE* bit to 1b.

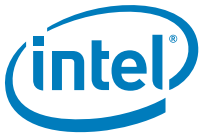
Note: If operation during $D3_{cold}$ is required, even when wake capability is not required (such as for manageability operation), the system should also set the *Auxiliary (AUX) Power PM Enable* bit in the PCIe Device Control register.

Transition to D3 for the case that the network link is needed: The network link is "considered" as needed if either wake up is enabled (either APM wake up or ACPI wake up) or proxy offload is enabled or the CSME requires the link. In this case, Foxville maintain the link. If the MANC.Keep_PHY_Link_Up bit (Veto bit) is cleared, the PHY can auto-negotiate to a lower link speed (for saving power) according to the device settings.

- **Transition to D3 for the case that the network link is not needed:** This case is taken if all the conditions above that define a "link is needed" are not metIf the "PHY Power Down Ena" flag in the CTRL_EXT register is active then Foxville sets its internal PHY to a power down state.
- Else, if the "LinkOff_EN_InAct" flag in the NVM is active then Foxville disconnects the Ethernet link.
- Else, same flow as if the link is needed.

5.2.3.2 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 D3cold state). To transition back to D0u, the system writes a 00b to the *Power State* field of the Power Management Control/Status Register (PMCSR). Transition to Dr state is through *PE_RST_N* assertion.



The *No_Soft_Reset* bit in the PMCSR register in Foxville is set to 1b, to indicate that Foxville does not perform an internal reset on transition from D3hot to D0 so that transition does not disrupt the proper operation of other active functions. In this case, software is not required to re-initialize the function's configuration space after a transition from D3hot to D0 (the function is in the $D0_{initialized}$ state); however, the software device driver needs to re-initialize internal registers since transition from D3hot to D0 causes an internal software reset (similar to asserting the software reset).

Foxville can also be configured via NVM to clear the *No_Soft_Reset* bit in the PMCSR register (see Section 6.1.1.27). In this case, an internal reset is generated when transition from D3hot to D0 occurs and functional context is not maintained also in PCIe configuration bits (except for bits defined as sticky). In this case, software is required to fully re-initialize the function after a transition to D0 as the Function is in the $D0_{uninitialized}$ state.

Note: The function is reset if the link state has made a transition to the L2/L3 ready state, on transition from D3cold to D0, if FLR is asserted or if transition D3hot to D0 is caused by assertion of PCIe reset (PE_RST pin) regardless of the value of the *No_Soft_Reset* bit.

5.2.3.3 Master Disable Via CTRL Register

System software can disable master accesses on the PCIe link by either clearing the PCI *Bus Master* bit or by bringing the function into a D3 state. From that time on, Foxville must not issue master accesses. Due to the full-duplex nature of PCIe, and the pipelined design in Foxville, it might happen that multiple requests are pending when the master disable request arrives. The protocol described in this section insures that a function does not issue master requests to the PCIe link after its *Master Enable* bit is cleared (or after entry to D3 state).

Two configuration bits are provided for the handshake between Foxville function and its software device driver:

- *GIO Master Disable* bit in the Device Control (CTRL) register - When the *GIO Master Disable* bit is set, Foxville blocks new master requests by this function. Foxville then proceeds to issue any pending requests by this function. This bit is cleared on master reset (LAN_PWR_GOOD, PCIe reset and software reset) to enable master accesses.
- *GIO Master Enable Status* bit in the Device Status (STATUS) register - Cleared by Foxville when the *GIO Master Disable* bit is set and no master requests are pending and is set otherwise. Indicates that no master requests are issued by this function as long as the *GIO Master Disable* bit is set. The following activities must end before Foxville clears the *GIO Master Enable Status* bit:
  - Master requests by the transmit and receive engines (for both data and MSI/MSI-X interrupts).
  - All pending completions to Foxville are received.

In the event of a PCIe Master disable (Configuration *Command register.BME* set to 0b) or LAN port disabled or if the function is moved into D3 state during a DMA access, Foxville generates an internal reset to the function and stops all DMA accesses and interrupts. Following a move to normal operating mode, the software device driver should re-initialize the receive and transmit queues of the relevant port.

Notes: The software device driver sets the *GIO Master Disable* bit when notified of a pending master disable (or D3 entry). Foxville then blocks new requests and proceeds to issue any pending requests by this function. The software device driver then polls the *GIO Master Enable Status* bit. Once the bit is cleared, it is guaranteed that no requests are pending from this function. The software device driver might time out if the *GIO Master Enable Status* bit is not cleared within a given time.

The *GIO Master Disable* bit must be cleared to enable a master request to the PCIe link. This can be done either through reset or by the software device driver.

5.2.4 Dr State (D3cold)

Transition to Dr state is initiated on several cases:

- On system power up - Dr state begins with the assertion of the internal power detection circuit and ends with de-assertion of *PE_RST_N*.
- On transition from a D0 state - During operation the system might assert *PE_RST_N* at any time. In an ACPI system, a system transition to the G2/S5 state causes a transition from D0 to Dr state.
- On transition from a D3 state - The system transitions Foxville into the Dr state by asserting PCIe *PE_RST_N*.

Any wake-up filter settings or proxying filter settings that were enabled before entering this reset state are maintained.

The system might 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 Dr state, Foxville might enter one of several modes with different levels of functionality and power consumption. The lower-power modes are achieved when Foxville is not required to maintain any functionality (see Section 5.6.1.1).

5.2.4.1 Entry to Dr State

Dr entry on platform power-up begins with the assertion of the internal power detection circuit. The NVM is read and determines Foxville configuration. If the *APM Enable* bit in the NVM's *Initialization Control Word 3* is set, then APM wake up is enabled. The PCIe link is not enabled in Dr state following system power up (since *PE_RST_N* is asserted).

Entering Dr state from D0a state is done by asserting *PE_RST_N*. An ACPI transition to the G2/S5 state is reflected in Foxville transition from D0a to Dr state. The transition can be orderly (such as user selecting the shut down option), in which case the software device driver might have a chance to intervene. Or, it might be an emergency transition (such as power button override), in which case, the software device driver is not notified.

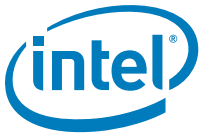
Transition from D3 (hot) state to Dr state is done by asserting *PE_RST_N*. Prior to that, the system initiates a transition of the PCIe link from L1 state to either the L2 or L3 state (assuming all functions were already in D3 state). The link enters L2 state if PCI-PM PME is enabled.

Following the transition to Dr state from the D0 state or from the D3 state, the WUC.APME and WUC.APMPME flags are programmed. These flags are programmed by the value of the "Dx APM PME# Enable" flag in the NVM. If it is set, then wake on magic packets is enabled **regardless** of the ACPI programming by the OS and its enabling in the PCIe configuration space.

Transition to Dr for the case that the network link is needed: The network link is "considered" as needed if either wake up is enabled (either APM wake up or ACPI wake up) or proxy offload is enabled or the CSME requires the link. In this case, Foxville maintain the link. If the MANC.Keep_PHY_Link_Up bit (Veto bit) is cleared, the PHY can auto-negotiate to a lower link speed (for saving power) according to the device settings. This step is done only if it was not already done in the transition to D3.

Transition to Dr for the case that the network link is not needed: This case is taken if all the above conditions that define a "link is needed" are not metRCV_TCO_EN.

- If Ultra Low Power state (ULP) is enable when entering the Dr state (by the "ULP_EN_Dr" flag in the NVM), then Foxville enters the ULP. See detailed description in Section 5.6.1.1.
- Else, if the "PHY Power Down Ena" flag in the CTRL_EXT register is active then Foxville sets its internal PHY to a power down state.



- Else, if the "LinkOff_EN_InAct" flag in the NVM is active then Foxville disconnects the Ethernet link.
- Else, same flow as if the link is needed.

5.2.4.2 Auxiliary Power Usage

The NVM *D3COLD_WAKEUP_ADVEN* bit and the *AUX_PWR* strapping pin determine when D3cold PME is supported:

- D3COLD_WAKEUP_ADVEN denotes that PME wake should be supported
- AUX_PWR strapping pin indicates that auxiliary power is provided

D3cold PME is supported as follows:

- If the *D3COLD_WAKEUP_ADVEN* is set to 1b and the AUX_PWR strapping is set to 1b, then *D3cold PME* is supported
- Else *D3cold PME* is not supported

The amount of power required for the function (including the entire NIC) is advertised in the Power Management Data register, which is loaded from the NVM.

If D3cold is supported, the *PME_En* and *PME_Status* bits of the PMCSR, as well as their shadow bits in the Wake Up Control (WUC) register are reset only by the power-up reset (detection of power rising).

5.2.5 Internal PHY state vs. Device State

The internal PHY is "aware" of the device power management state as well as the "static" device enablement option by the LAN_DISABLE_N input signal.

5.2.5.1 Link Speed vs. Device Power State

Foxville supports an option to set the link speed that are advertised by the internal PHY according to the device power state. This option is aimed to limit the power consumption of the device due to system requirements. It is enabled by the following *PHPM* register settings (loaded from the NVM):

- *Disable 2500* - disable 2.5 Gb/s altogether
- *Disable 2500 in non-D0a* - disable 2.5 Gb/s when the device is in non-D0a states
- *Disable 1000* - disable 1 Gb/s altogether
- *Disable 1000 in non-D0a* - disable 1 Gb/s when the device is in non-D0a states
- *Disable 100 in non-D0a* - disable 100 Mb/s when the device is in non-D0a states

Updating the "*Disable 2500*" or "*Disable 1000*" setting options Foxville restart link AN with the updated settings. It does so if not inhibited by the CSME (by setting the KEEP_PHY_LINK_UP flag in the MANC register). Same flow happens if the device power state is changed and the setting option for D0a and none-D0a differ.

5.2.5.2 PHY State vs. Device Power State

Foxville supports an option to disconnect the Ethernet link or set the whole PHY unit to a power down state according to the device state. These options are enabled by the "PHY Link Down Ena" and "PHY Power Down Ena" flags in the CTRL_EXT register. See "Entry to D3 State" and "Entry to Dr State" sections for a description when the PHY enters these saving power states.

5.2.5.3 PHY State vs. Static Device Enable Option

Driving the LAN_DISABLE_N signal the whole device or only the PHY enters conditionally to one of the following power saving modes:

- If ULP is enabled then Foxville enters the Ultra Low Power state as detailed in Section 5.6.1.1.
- Else, if the "PHY Power Down Ena" flag in the CTRL_EXT register is active then the internal PHY is set to a power down state.
- Else, if the "PHY Link Down Ena" flag in the CTRL_EXT register is active then Foxville disconnects the Ethernet link.

5.2.6 Link Disconnect

When Foxville detects that the link is disconnected (by the absence of NLP and FLP pulses) it indicates no link in the "*Link Energy Detect*" flag in the PHPM register. In this case the Link Up indication in the STATUS register (bit 1) is inactive as well.

5.2.7 Device Off States

5.2.7.1 Static Device Off

Foxville enters a global power-down state initiated by the BIOS when the LAN_DISABLE_N pin is asserted. See Section 5.6, Section 5.6.1 and Section 5.6.1.1.

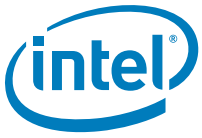
5.2.7.2 Dynamic Device Off

Foxville enters a global power-down state initiated by the OS as detailed in Section 5.6.1.1.

5.3 Power Limits by Certain Form Factors

Foxville exceeds the allocated auxiliary power in some configurations. Foxville must therefore be configured to meet the previously mentioned requirements. To do so, Foxville implements three Flash bits to disable operation in certain cases:

1. The *PHPM.Dis2500* PHY register bit disables 2.5 Gb/s operation.
2. The *PHPM.Dis2500_nonD0a* PHY CSR bit disables 2.5 Gb/s operation in non-D0a states. If *PHPM.Dis2500_nonD0a* is set, and Foxville is at 2.5Gb/s speed in the D0a state, then on a transition to non-D0a state Foxville removes advertisement for 2.5 Gb/s and re-auto-negotiates the link speed.
3. The *PHPM.Dis1000* PHY register bit disables 1000 Mb/s and 2.5 Gb/s operation under all conditions.
4. The *PHPM.Dis1000_nonD0a* PHY CSR bit disables 1000 Mb/s and 2.5 Gb/s operation in non-D0a states. If *PHPM.Dis1000_nonD0a* is set, and Foxville is at 1000 Mb/s or 2.5 Gb/s speed on entry to a non-D0a state, then Foxville removes advertisement for 1000 Mb/s and auto-negotiates.
5. The *PHPM.Dis100_nonD0a* PHY CSR bit disables 1000 Mb/s and 100 Mb/s operation in non-D0a states. If *PHPM.Dis100_nonD0a* is set, and Foxville is at 1000 Mb/s or 100 Mb/s speeds on entry to a non-D0a state, then Foxville removes advertisement for 1000 Mb/s and 100 Mb/s and auto-negotiates.



Note that Foxville restarts link auto-negotiation each time it transitions from a state where a specific speed is enabled to a state where this speed is disabled, or vice versa. For example, if *PHPM.Dis1000_nonD0a* is set but *PHPM.Dis1000* is cleared, and the link is established at 1G b/s in D0a and Foxville transits from D0a state to D3 or Dr states then it triggers autonomously link auto-negotiation.

5.4 Interconnects Power Management

This section describes the power reduction techniques employed by Foxville main interconnects.

5.4.1 PCIe Link Power Management

Foxville supports all PCIe power management link states:

- L0 state is used in D0u and D0a states.
- The L1 state and L1 sub-states are also used in D0a and D0u states when idle conditions apply for a longer period of time. The L1 state is also used 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 if *PME* is not enabled in other Dr transitions.

Foxville support for active state link power management is reported via the PCIe *Active State Link PM Support* register and is loaded from the NVM.

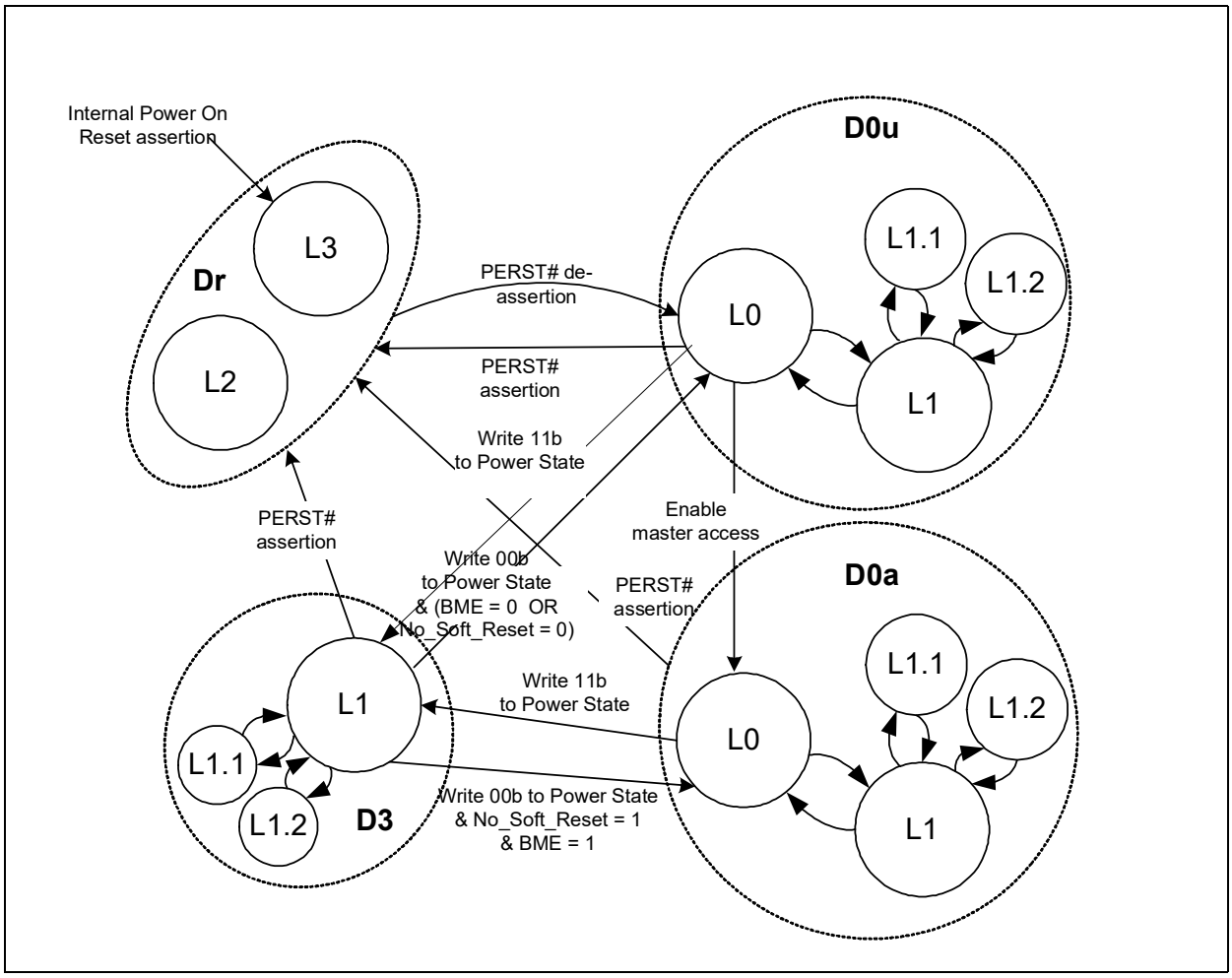


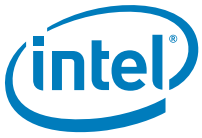
Figure 5-2. Link Power Management State Diagram

The following NVM fields control L1 behavior:

- *Act_Stat_PM_Sup* - Indicates support for ASPM L1 in the PCIe configuration space (loaded into the Active State Link PM Support field)
- *L1_Act_Ext_Latency* - Defines L1 active exit latency
- *L1_Act_Acc_Latency* - Defines L1 active acceptable exit latency
- *Latency_To_Enter_L1* - Defines the period before the transition into L1 state (if the *PCIEMISC.Lx_decision* bit is set to 0b)

5.5 Wake Up

Foxville supports two modes of wake-up management:



1. Advanced Power Management (APM) wake up
2. ACPI/PCIe defined wake up

The usual model is to activate one mode at a time but not both modes together. If both modes are activated, Foxville might wake up the system on unexpected events. For example, if APM is enabled together with the ACPI/PCIe Magic packet in the *WUFC* register, a magic packet might wake up the system even if APM is disabled (*WUC.APME* = 0b). Alternatively, if APM is enabled together with some of the ACPI/PCIe filters (enabled in the *WUFC* register), packets matching these filters might wake up the system even if PCIe PME is disabled.

5.5.1 Advanced Power Management Wake Up

Advanced Power Management Wake Up or APM Wakeup (also known as Wake on LAN) is a feature that existed in earlier 10/100 Mb/s NICs. This functionality was designed to receive a broadcast or unicast packet with an explicit data pattern, and then assert a subsequent signal to wake up the system. This was accomplished by using a special signal that ran across a cable to a defined connector on the motherboard. The NIC would assert the signal for approximately 50 ms to signal a wake up. Foxville now uses (if configured) an in-band PM_PME message for this functionality.

On power up, Foxville reads the *APM Enable* bits from the NVM *Initialization Control Word 3* into the *APM Enable (APME)* bits of the *Wakeup Control (*WUC*)* register. These bits control enabling of APM wake up.

When APM wake up is enabled, Foxville checks all incoming packets for Magic packets. See Section 5.5.3.1.4 for a definition of Magic packets.

Once Foxville receives a matching Magic packet, and if the *WUC.APMPME* bit or the *PMCSR.PME_En* bits are set to 1b and the *WUC.APME* bit is set to 1b it:

- Sets the *PME_Status* bit in the *PMCSR* register and issues a PM_PME message (in some cases, this might require asserting the PE_WAKE_N signal first to resume power and clock to the PCIe interface).
- Stores the first 1500 bytes of the packet in the Wake Up Packet Memory (WUPM and WUPM_EXT registers).
- Sets the *Magic Packet Received* bit in the Wake Up Status (WUS) register.
- Sets the packet length in the Wake Up Packet Length (WUPL) register.

Foxville maintains the first Magic packet received in the *Wake Up Packet Memory* (WUPM) register until the software device driver writes a 1b to the *WUS.MAG* bit.

If the *WUC.EN_APM_D0* bit is set to 1b, APM wake up is supported in all power states and only disabled if a subsequent NVM read results in the *WUC.APME* bit being cleared or software explicitly writes a 0b to the *WUC.APME* bit. If the *WUC.EN_APM_D0* bit is cleared APM wake-up is supported only in the D3 or Dr power states.

Notes:

1. When the *WUC.APMPME* 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 *WUC.APMPME* bit after power-up or PCIe reset.
2. If APM is enabled and Foxville is programmed to issue a wake event on the PCIe, each time a Magic packet is received, a wake event is generated on the PCIe interface even if the *WUS.MAG* bit was set as a result of reception of a previous Magic packet. Consecutive magic packets generate consecutive Wake events.

5.5.2 ACPI Power Management Wake Up

Foxville supports PCIe power management based wake-up. It can generate system wake-up events from a number of sources:

- Reception of a Magic packet.
- Reception of a network wake-up packet.
- Detection of a change in network link state (cable connected or disconnected).
- Wake-up by manageability after receiving an unsupported packets for proxying.

Activating PCIe power management wake up requires the following:

- System software writes at configuration time a 1b to the PCI *PMCSR.PME_En* bit.
- Software device driver clears all pending wake-up status bits in the Wake Up Status (WUS and WUS_EXT) registers.
- The software device driver programs the Wake Up Filter Control (WUFC) register to indicate the packets that should initiate system wake up and programs the necessary data to the IPv4/v6 Address Table (*IP4AT, IP6AT*) and the Flexible Host Filter Table (FHFT and FHFT_EXT). It can also set the *WUFC.LNKC* bit to cause wake up on link status change.
- Once Foxville wakes the system, the software device driver needs to clear the WUS, WUS_EXT, WUFC and WUFC_EXT registers until the next time the system moves to a low power state with wake up enabled.

Normally, after enabling wake up, system software moves the device to D3 low power state by writing a 11b to the PCI *PMCSR.Power State* field.

Once wake up is enabled, Foxville monitors incoming packets, first filtering them according to its standard address filtering method, then filtering them with all of the enabled wake-up filters. If a packet passes both the standard address filtering and at least one of the enabled wake-up filters, Foxville:

- Sets the *PME_Status* bit in the PMCSR.
- Asserts PE_WAKE_N (if the *PME_En* bit in the PMCSR configuration register is set).
- Stores the first 1500 bytes of the packet in the Wakeup Packet Memory (WUPM and WUPM_EXT registers).
- Sets one or more bits in the Wake Up Status (WUS / WUS_EXT) register. Note that Foxville sets more than one bit if a packet matches more than one filter.
- Sets the packet length in the Wake Up Packet Length (WUPL) register.

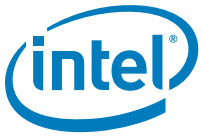
Note: If enabled, a link state change wake-up causes similar results. Sets the *PMCSR.PME_Status* bit, asserts the *PE_WAKE_N* signal and sets the relevant bit in the WUS register.

The PE_WAKE_N remains asserted until the operating system either writes a 1b to the *PMCSR.PME_Status* bit or writes a 0b to the *PMCSR.PME_En* bit.

After receiving a wake-up packet, Foxville ignores any subsequent wake-up packets until the software device driver clears all of the received bits in the Wake Up Status (WUS / WUS_EXT) register. It also ignores link change events until the software device driver clears the *Link Status Changed* (*LNKC*) bit in the Wake Up Status (WUS) register.

5.5.3 Wake-Up and Proxying Filters

Foxville supports issuing wake-up to Host when device is in D3 or protocol offload (proxying) of packets using two types of filters:



- Pre-defined filters
- Flexible filters

Each of these filters are enabled if the corresponding bit in the Wake Up Filter Control (WUFC) register or Proxying Filter Control (PROXYFC) register is set to 1b.

Note: When VLAN filtering is enabled, packets that passed any of the receive wake-up filters should only cause a wake-up event if they also passed the VLAN filtering.

5.5.3.1 Pre-Defined Filters

The following packets are supported by Foxville's pre-defined filters:

- Directed packet (including exact, multicast indexed, and broadcast)
- Magic Packet
- ARP/IPv4 request packet
- Directed IPv4 packet
- Directed IPv6 packet
- ICMPv6 packet like:
  - — IPv6 Neighbor Solicitation (NS) packet
  - — IPV6 Multicast Listener Discovery (MLD) packet

Each of these filters are enabled if the corresponding bit in the *Wakeup Filter Control (WUFC)* register or *Proxying Filter Control* (*PROXYFC*) register is set to 1b.

Following sections include a description of each filter and a table describing which bytes at which offsets need to be compared, to determine if the packet passes the filter.

Note: Both VLAN fields and LLC/SNAP fields can increase the given offsets if they are present.

5.5.3.1.1 Directed Exact Packet

Foxville generates a wake-up event after receiving any packet whose destination address matches one of the 16 valid programmed receive destination MAC addresses (defined in the *RAL* and *RAH* registers), if the *Directed Exact Wake Up Enable* bit is set in the *Wake Up Filter Control* (*WUFC.EX*) register.

Foxville forwards a packet to Management for Proxying after receiving any packet whose MAC destination address matches one of the 16 valid programmed receive destination MAC addresses (defined in the *RAL* and *RAH* registers), if the *Directed Exact Proxying Enable* bit is set in the *Proxying Filter Control* (*PROXYFC*) register.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	Match any pre-programmed address

5.5.3.1.2 Directed Multicast Packet

For multicast packets, the upper bits of the incoming packet's destination address index a bit vector, the Multicast Table Array (*MTA*) that indicates whether to accept the packet.

If the *Directed Multicast Wake Up Enable* bit set in the *Wake Up Filter Control* (*WUFC.MC*) register and the indexed bit in the vector is one, then Foxville generates a wake-up event. If the *Directed Multicast Proxying Enable* bit is set in the *Proxying Filter Control* (*PROXYFC*) register and the indexed bit in the vector is one, then Foxville forwards packet to Management for Proxying.

Note: The exact bits used in the comparison are programmed by software in the *Multicast Offset* field of the Receive Control (*RCTL.MO*) register.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	See Section 5.5.3.1.2.

5.5.3.1.3 Broadcast

If the Broadcast Wake Up Enable bit in the Wake Up Filter Control (WUFC.BC) register is set, Foxville generates a wake-up event when it receives a broadcast packet. If the Broadcast Proxying Enable bit in the Proxying Filter Control (PROXYFC) register is set, Foxville forwards packet to Management for Proxying when receiving a broadcast packet.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address	FF*6	Compare	

5.5.3.1.4 Magic Packet

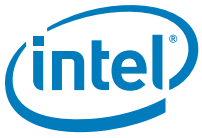
Once the LAN controller has been put into the Magic Packet mode, it scans all incoming frames addressed to the node for a specific data sequence. This sequence indicates to the controller that this is a Magic Packet frame. A Magic Packet frame must also meet the basic requirements for the LAN technology 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. 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."

Foxville expects the destination address to either:

- Be the broadcast address (FF.FF.FF.FF.FF.FF)
- Match the value in Receive Address 0 (*RAH0, RAL0*) register. This is initially loaded from the NVM but can be changed by the software device driver.
- Match any other address filtering (*RAH*[n], *RAL*[n]) enabled by the software device driver.

Foxville searches for the contents of Receive Address 0 (*RAH0, RAL0*) register as the embedded IEEE address. It considers any non-0xFF byte after a series of at least 6 0xFFs to be the start of the IEEE address for comparison purposes. For example, it catches the case of 7 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 6 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 the *Broadcast Accept* bit of the *Receive Control (RCTL.BAM)* register is 0b. If APM wake up (wake up by a Magic



Packet) is enabled in the NVM, Foxville starts up with the Receive Address 0 (*RAH0, RAL0*) register loaded from the NVM. This enables Foxville 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	S=(0/4/8)	Possible VLAN Tags (single or double)		Skip	
12 + S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + 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 Receive Address 0 (RAH0, RAL0) register.

5.5.3.1.5 ARP/IPv4 Request Packet

Foxville supports receiving ARP request packets for wake up if the *Directed ARP* bit or the *ARP* bit is set in the *Wake Up Filter Control* (*WUFC*) register and Proxying if the *Directed ARP* bit or the *ARP* bit is set in the *Proxying Filter Control* (*PROXYFC*) register.

- If the *Directed ARP* bit is set, a successfully matched packet must contain a broadcast MAC address, match VLAN tag if programmed, a Ethernet type of 0x0806, an ARP op-code of 0x01 and the Target IP address matches one of the four IPv4 addresses programmed in the *IPv4 Address Table* (*IP4AT*).
- If the *ARP* bit is set, a successfully matched packet must contain a broadcast MAC address, match VLAN tag if programmed, a Ethernet type of 0x0806 and an ARP op-code of 0x01.

Foxville also handles ARP request packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Table 5-3. ARP Packet Structure and Processing

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	S=(0/4/8)	Possible VLAN Tags (single or double)		Compare on internal VLAN only	Processed by main address filter.
12 + S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Ethernet Type	0x0806	Compare	ARP
14 + S + D	2	HW Type	0x0001	Compare	
16 + S + D	2	Protocol Type	0x0800	Compare	
18 + S + D	1	Hardware Size	0x06	Compare	
19 + S + D	1	Protocol Address Length	0x04	Compare	
20 + S + D	2	Operation	0x0001	Compare	
22 + S + D	6	Sender HW Address	-	Ignore	

Table 5-3. ARP Packet Structure and Processing

Offset	# of bytes	Field	Value	Action	Comment
28 + S + D	4	Sender IP Address	-	Ignore	
32 + S + D	6	Target HW Address	-	Ignore	
38 + S + D	4	Target IP Address	IP4AT	Compare	Compare if the *Directed ARP* bit is set to 1b. May match any of four values in *IP4AT*.

5.5.3.1.6 Directed IPv4 Packet

Foxville supports receiving directed IPv4 packets for wake up if the *IPV4* bit is set in the Wake Up Filter Control (*WUFC*) register and Proxying if the *IPV4* bit is set in the *Proxying Filter Control* (*PROXYFC*) register.

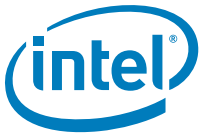
Four IPv4 addresses are supported, which are programmed in the IPv4 Address Table (*IP4AT*). A successfully matched packet must contain the station's MAC address, match VLAN tag if programmed, a Ethernet type of 0x0800, and one of the four programmed IPv4 addresses. Foxville also handles directed IPv4 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Table 5-4. IPv4 Packet Structure and Processing

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	S=(0/4/8)	Possible VLAN Tags (single or double)		Compare on internal VLAN only	Processed by main address filter.
12 + S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Ethernet Type	0x0800	Compare	IPv4
14 + S + D	1	Version/ HDR length	0x4X	Compare	Check IPv4
15 + S + D	1	Type of Service	-	Ignore	
16 + S + D	2	Packet Length	-	Ignore	
18 + S + D	2	Identification	-	Ignore	
20 + S + D	2	Fragment Info	-	Ignore	
22 + S + D	1	Time to live	-	Ignore	
23 + S + D	1	Protocol	-	Ignore	
24 + S + D	2	Header Checksum	-	Ignore	
26 + S + D	4	Source IP Address	-	Ignore	
30 + S + D	4	Destination IP Address	IP4AT	Compare	May match any of four values in *IP4AT*.

5.5.3.1.7 Directed IPv6 Packet

Foxville supports receiving directed IPv6 packets for wake up if the *IPV6* bit is set in the *Wake Up Filter Control (WUFC)* register and Proxying if the *IPV6* bit is set in the *Proxying Filter Control* (*PROXYFC*) register.



One IPv6 address is supported and is programmed in the *IPv6 Address Table (IP6AT)*. A successfully matched packet must contain the station's MAC address, match VLAN tag if programmed, a Ethernet type of 0x86DD, and the programmed IPv6 address. In addition, the *IPAV.V60* bit should be set. Foxville also handles directed IPv6 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Table 5-5. IPv6 Packet Structure and Processing

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	S=(0/4/8)	Possible VLAN Tags (single or double)		Compare on internal VLAN only	Processed by main address filter.
12+ S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Ethernet Type	0x86DD	Compare	IPv6
14 + S + D	1	Version/ Priority	0x6X	Compare	Check IPv6
15 + S + D	3	Flow Label	-	Ignore	
18 + S + D	2	Payload Length	-	Ignore	
20 + S + D	1	Next Header	-	Ignore	
21 + S + D	1	Hop Limit	-	Ignore	
22 + S + D	16	Source IP Address	-	Ignore	
38 + S + D	16	Destination IP Address	IP6AT	Compare	Match value in *IP6AT*.

5.5.3.1.8 Neighbor Solicitation (NS) and Multicast Listener Discovery (MLD) IPv6 Packets

Foxville supports receiving:

1. IPv6 Neighbor Solicitation (NS) packets sent by a node to determine the link-layer address of a neighbor, or to verify that a neighbor is still reachable for wake up or Proxying.
2. IPV6 Multicast Listener Discovery (MLD) packets sent by an IPv6 router to discover the presence of multicast listeners (that is, nodes wishing to receive multicast packets) on its directly attached links, and to discover specifically which multicast addresses are of interest to those neighboring nodes.
3. Other ICMPv6 packets.

If the *NS* or *NS Directed* bits are set in the *Wake Up Filter Control (WUFC)* register, Wake up is executed on reception of the relevant ICMPv6 packets. Else if the *NS* or *NS Directed* bits are set in the *Proxying Filter Control* (*PROXYFC*) register the relevant ICMPv6 packets are sent to Firmware for Protocol offload.

- If the *NS directed* bit is set a successfully matched packet must contain the station's MAC address (Unicast or Multicast), match VLAN tag if programmed, a Ethernet type of 0x86DD, a IPv6 Header Type of ICMPv6 (0x3A), correct ICMPv6 Checksum and the single programmed IPv6 address in the *IPv6 Address Table* (*IP6AT*) must match the Target IPv6 Address in a NS packet or the Multicast Address field in a MLD packet. In addition, the *IPAV.V60* bit should be set.
- If the *NS* bit is set a successfully matched packet must contain the station's MAC address (Unicast or Multicast), match VLAN tag if programmed, a protocol type of 0x86DD, a IPv6 Header Type of ICMPv6 (0x3A) and a correct ICMPv6 Checksum. In this case all ICMPv6 packets are forwarded to Firmware.

Foxville also handles Neighbor Solicitation (NS) and Multicast Listener Discovery (MLD) IPv6 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Table 5-6. Neighbor Solicitation (NS) and Multicast Listener Discovery (MLD) Packet Structure and Processing

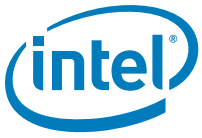
Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC header – processed by main address filter.
6	6	Source Address		Skip	
12	S=(0/4/8)	Possible VLAN Tags (single or double)		Compare on internal VLAN only	Processed by main address filter.
12+ S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Ethernet Type	0x86DD	Compare	IPv6
14 + S + D	1	Version/ Priority	0x6X	Compare	Check IPv6
15 + S + D	3	Flow Label	-	Ignore	
18 + S + D	2	Payload Length	-	Ignore	
20 + S + D	1	Next Header	IPv6 next header types or 0x3A	Compare	58 decimal (0x3A) - ICMPv6 header type
21 + S + D	1	Hop Limit	-	Ignore	
22 + S + D	16	Source IP Address	-	Ignore	
38 + S + D	16	Destination IP Address	-	Ignore	
54+D+S	N	Possible IPv6 Next Headers	-	Ignore	
ICMPv6 header					
54+D+S+N	1	Type	-	Ignore	
55+D+S+N	1	Code	0x0	Ignore	
56+D+S+N	2	Checksum		Check	
58+D+S+N	4	Reserved	0x0	Ignore	
62+D+S+N	16	Target IP Address/ Multicast Address	IP6AT	Compare	Match value in *IP6AT* for *NS directed* Match. **Note:** Relevant for NS and MLD packets.
78+D+S+N	F	ICMPv6 Message Body		Ignore	

5.5.3.2 Flexible Filters

Foxville supports a total of 32 flexible filters. Each filter can be configured to recognize any arbitrary pattern within the first 128 bytes of the packet. To configure the flexible filters, software programs the mask values (required values and the minimum packet length), into the Flexible Host Filter Table (*FHFT* and *FHFT_EXT together with the FHFTSL register*). These 32 flexible filters contain separate values for each filter.

To enable Wake on LAN operation based on the Flex filters Software must also enable the filters in the *Wake Up Filter Control* (*WUFC*) register, and enable the overall wake up functionality. The overall wake up functionality must be enabled by setting *PME_En* bit in the *PMCSR* configuration register or the *PME_En* bit in the *Wake Up Control* (*WUC*) register.

To enable Proxying operation based on the Flex filters Software must also enable the filters in the *Proxying Filter Control* (*PROXYFC*) register, and enable the overall Proxying functionality by setting to 1b the *WUC.PPROXYE* bit.



Once enabled, the flexible filters scan incoming packets for a match. If the filter encounters any byte in the packet where the mask bit is one and the byte doesn't match the value programmed in the Flexible Host Filter Table (*FHFT* or *FHFT_EXT*), then the filter fails that packet. If the filter reaches the required length without failing the packet, it passes the packet and generates a wake-up event. It ignores any mask bits set to one beyond the required length.

Note: The flex filters are temporarily disabled when read from or written to by the host. Any packet received during a read or write operation is dropped. Filter operation resumes once the read or write access completes.

The packets listed in the following subsections are listed for reference purposes only. The flexible filter could be used to filter these packets as well as other ones.

Flexible Filters and Preemption

If preemption is enabled and if more than the first 64 bytes are enabled in the flexible filters then the minimum preemption fragment size should be set to 128 bytes. For preemption functionality see Section 7.5.3.1.

5.5.3.2.1 IPX Diagnostic Responder Request Packet

An IPX diagnostic responder request packet must contain a valid MAC address, a Ethernet type of 0x8137, and an IPX diagnostic socket of 0x0456. It might include LLC/SNAP headers and VLAN tags. Since filtering this packet relies on the flexible filters, which use offsets specified by the operating system directly, the operating system must account for the extra offset LLC/SNAP headers and VLAN tags.

Table 5-7. IPX Diagnostic Responder Request Packet Structure and Processing

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	
6	6	Source Address		Skip	
12	S=(0/4/8)	Possible VLAN Tags (single or double)		Skip	
12+ S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Ethernet Type	0x8137	Compare	IPX
14 + S + D	16	Some IPX Stuff	-	Ignore	
30 + S + D	2	IPX Diagnostic Socket	0x0456	Compare	

5.5.3.2.2 Directed IPX Packet

A valid directed IPX packet contains the station's MAC address, a Ethernet type of 0x8137, and an IPX node address that is equal to the station's MAC address. It might include LLC/SNAP headers and VLAN tags. Since filtering this packet relies on the flexible filters, which use offsets specified by the operating system directly, the operating system must account for the extra offset LLC/SNAP headers and VLAN tags.

Table 5-8. IPX Packet Structure and Processing

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	S=(0/4/8)	Possible VLAN Tags (single or double)		Skip	
12+ S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Ethernet Type	0x8137	Compare	IPX
14 + S + D	10	Some IPX Info	-	Ignore	
24 + S + D	6	IPX Node Address	Receive Address 0	Compare	Must match receive address 0.

5.5.3.2.3 Utilizing Flex Wake-Up Filters In Normal Operation

Foxville enables utilizing the WoL Flex filters in normal operation, when in D0 power management state, for queuing decisions. Further information can be found in Section 7.1.2.5.

5.5.3.3 Wake Up Packet Storage

Foxville saves the first 1500 bytes of the wake-up packet in its internal buffer, which can be read through the *Wake Up Packet Memory (WUPM)* register after the system wakes up.

5.6 Ultra Low Power - ULP

Foxville support an ultra low power state on which the device consume its lowest power of ***~4mW***. The digital logic in Foxville operates at 0.9v provided by an internal DC2DC voltage regulator. Powering off the internal DC2DC converter shut down the digital logic of the device saving both active and leakage power. This state is called ULP state. During the ULP state Foxville maintain an "Always On Power" domain (AON domain) that preserve waking up functionality. As part of the wakeup options, Foxville is able to detect link energy. Foxville supports also the "Back to Back" functionality at ULP state enabling both sides of the link to be in the ULP state and still detect link energy. Foxville does so by sending periodic beacons (NLP's) during the ULP state at a pre-programmed rate in the "ULP Beacon Setting" word in the NVM.

While in the ULP state the PCIe bus is inactive, all digital IO pins are either behave as input signals or at high impedance. The Link signals are not driven other than possible beacon pulses if enabled.

5.6.1 Entering the ULP State

The Figure 5-3 below provides a conceptual description of the ULP logic entering the ULP state. Foxville enters the ULP state if all the following conditions are met:

- The ULP_Req flag in the PHPM register is set
- The "Inhibit ULP" flag in the MANC register is inactive (this flag can be set by the CSME).
- All enabled signals shown in figure Figure 5-3 for entering the ULP do not require the device to be active

Entering the ULP state, most of the logic is turned off losing most of the device settings. As such, critical device settings must be preserved before entering the ULP state.

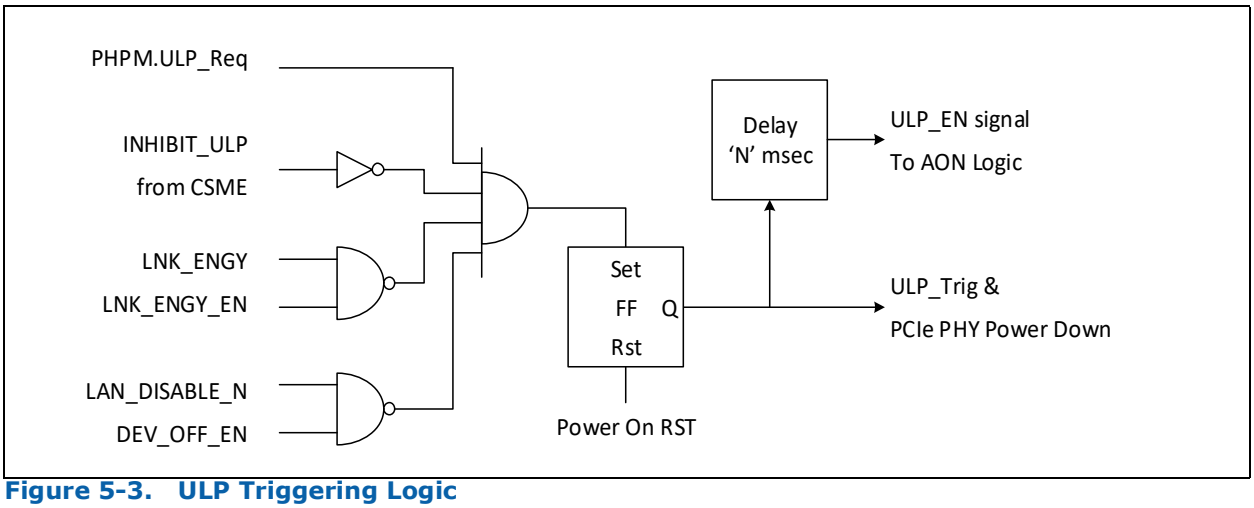


Figure 5-3. ULP Triggering Logic

5.6.1.1 The Flow Entering the ULP State initiated by the OS

The flow is initiated by one of two possible events: The OS sets Foxville to the D3 cold or following a link status change to inactive. There are several causes on which the OS sets Foxville to the D3 cold state: System transits to the Sx state or Run Time D3 state - RTD3 or "Dynamic Device Off". The flow as it is observed by Foxville is the same for all these cases:

1. The OS indicates to the LAN driver that it is about to transit Foxville to the Dx state. It also instruct the driver about the causes the device should wake up from the Dx state as well as the Proxy off-load functionality. As a result, the device driver program the required wakeup and proxy filters and enable the wakeup events in the WUC and WUFC registers.
2. The OS transits Foxville to the D3 and then possibly to the Dr state. This step can be part of the flow transiting to system Sx or RTD3 cold.
3. The OS also calls the BIOS _OFF method that will assert LAN_DISABLE_n.
4. Following the assertion of the LAN_DISABLE_n signal, the MNG engine gets an interrupt that in case of assertion of LAN_DISABLE_N will enable to exit the ULP by the LAN_DISABLE_n signal negation in the "PMU ULP Configuration" and the "PMU Beacon Configuration" Registers in the PHY TOP (offset 0x0 and offset 0x3 in base address 0xA300 respectively).
5. Following the transition to the Dr states,Foxville transits conditionally to the ULP state according to the following sequence:
   - If the device is not at Dr state then abort the ULP sequence. This step is needed for the case that the flow is initiated by link status change event to inactive.
   - If link was active prior to entering the D3 state, then Foxville might re-started Autonegotiation process at lower link speed when it transited to the D3 state. As such, the link is lost for "some" time until it is re-established. In order to overcome this transient state, Foxville waits ULP_Delay seconds before proceeding with the following steps (the ULP_Delay parameter is defined in the "ULP Capability Enable" word in the NVM).
   - Check that ULP at Dr state is enabled by the "ULP_EN_Dr" flag in the "ULP Capability Enable" word in the NVM. If ULP is not enabled then abort the ULP sequence.
   - If the link functionality is needed and the link is active then abort the ULP sequence. The link functionality is needed if at least one of the following flags is active: MANC->RCV_TCO_EN OR WUC->PPROXYE OR WUC->PME_En OR the WUC->APMPME.

- Set the causes to exit the ULP:
  - Enable to exit the ULP by the PE_CLKREQ_N signal.
  - Enable to exit the ULP by the ULP_WAKE_N signal if the CSME is identifies as active.
  - Enable to exit the ULP by link energy when the link functionality is required..

— If the CSME is active then initiate the "ULP Indication AEN" message to the CSME indicating that Foxville is about to enter the ULP state initiated by the OS. Then wait ~100usec to guarantee that the CSME got this message.

— Set the following flags in the PHPM register enabling the device to enter the ULP:

- Clear the DEV_OFF_EN and LNK_ENGYEN flags
- If the link functionality is required and beacon is enabled by the BCON_EN_Dr flag in the NVM, then set the SPD_B2B_EN flag in the PHPM register.
- Set the ULP_Req flag that triggers the ULP sequence.

— Once the Inhibit ULP is cleared (initiated by the CSME), the ULP_Trig flag in the PHPM register is set and after a programmable delay (in the PHPM register) the ULP_En signal to the always on logic is asserted and Foxville enters the ULP state.

Table 5-9. Conditions to Enter and Exit the ULP

OS: wake on or Proxy is active	CSME: present and Enabled	Link State	Enter ULP condition	Exit ULP condition
No	No	X	Enter Unconditional to ULP (Note that the "Inhibit ULP" is expected to be inactive)	Host access
X	Yes	Inactive	Conditional enter to ULP if the "Inhibit ULP" is inactive (*)	Host access or Link Energy or CSME access (if CSME is enabled)
X	Yes	Active	ULP sequence is aborted = no ULP	N/A
Yes	X	Inactive	Conditional enter to ULP if the "Inhibit ULP" is inactive (*)	Host access or Link Energy or CSME access (if CSME is enabled)
Yes	X	Active	ULP sequence is aborted = no ULP	N/A
(*) Note that the CSME clears the "Inhibit ULP" when the link goes inactive				

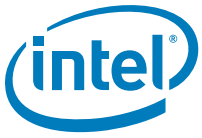
5.6.2 Exiting the ULP State

Foxville exits the ULP state if at least one of the enabled signals to exit the ULP require an active device:

{ (Dev_OFF_EN & LAN_DISABLE_n) | (PE_CLKREQ_EN & ~PE_CLKREQ_n) | (ULP_WAKE_EN & ~ULP_WAKE_n) | (PHY_Energy_EN & PHY_Energy) }

Exiting the ULP state, Foxville gets its power back and it is initialized as if LAN_PWR_Good goes high:

- Foxville checks if it woke-up from the ULP state or it is a simple power up.
  — If the device woke-up from the ULP state and if it woke-up by link energy or by the CSME driving the ULP_WAKE_n signal then assert the PE_WAKE_n to the host.
- Initiate a discovery notify message to the CSME on the SMBus. Once the PCIe bus is active, initiate a "Media Change" AEN message to the CSME on the SMBus (indicating that the PCIe bus is available as well).



5.6.2.1 Exiting ULP by Host

On exit from RTD3, the OS calls the BIOS _ON method which can be assigned to de-assert LAN_DISBALE_N pin using GPIO and satisfy the equation for exiting ULP.

5.6.2.2 Exiting ULP by CSME

The CSME can trigger ULP exit by asserting ULP_WAKE_N pin. This can be done by connecting SMB_DATA to ULP_WAKE_N.

5.7 Reset On LAN (RoL)

Low cost systems without a BMC sometimes require remote boot capability. Foxville supports generation of a system level reset via assertion the PCIe WAKE# pin on reception of a pre defined packet, for systems that do not require any WoL capability and do not enable PCIe OBFF. Foxville also supports remote boot capability for systems that require the PCIe WAKE# pin for WoL or PCIe OBFF by issuing a system level reset on reception of a pre defined packet via a dedicated SDP pin when the WUC.SRST_PIN_EN register bit is set.

To support RoL the Foxville should be programmed by the driver in the following manner:

1. WUC.PME_En bit should be set to 1b to enable wake up functionality
2. Program the Flexible Host Filter Table (FHFT), Data, Mask and length with the RoL packet
3. Enable the flex filter to RoL by enabling the flex filter and setting the filter's action bit to RoL through setting the appropriate bits in WUFC
4. The WUC.SRST_PIN_EN bit should be set to 1b to enable issuing a system reset on reception of a matched wake up packet via a SDP pin. The SDP pin chosen for this functionality needs to be defined in the WUC.SRST_PIN_SEL field
5. Two modes of RoL are defined by the WUC.RoL_Mode when set to 0b a 50mSec pulse will be driven by the SDP, when set to 1b the RoL event will drive a state change in the SDP and it is the driver responsibility to clear the pin assertion post reset - clearing the pin is done by setting the proper values in the WUC.SRST_SET and WUC.SRST_VAL.

5.8 Protocol Offload (Proxying)

In order to avoid spurious wake-up events and reduce system power consumption when the device is in D3 low power state and system is in S3 or S4 low power states, Foxville supports protocol offload (proxying) of:

1. A single IPv4 Address Resolution Protocol (ARP) request.
   - Responds to IPv4 address resolution request with the host MAC (L2) address (as defined in RFC 826).
2. Two IPv6 Neighbor Solicitation (NS) requests, where each NS protocol offload request includes two IPv6 addresses, for a total of four possible IPv6 addresses.
   - IPv6 NS requests with the host MAC (L2) address (as defined in RFC 4861).
3. When NS protocol offload is enabled, Foxville supports up to two IPv6 Multicast-Address-Specific Multicast Listener Discovery (MLD) queries (either MLDv1 or MLDv2). In addition, Foxville also responds to general MLD queries, used to learn which IPv6 multicast addresses have listeners on an attached link.
   - MLD protocol offload is supported when NS protocol offload is enabled so that IPv6 routers discover the presence of multicast listeners (that is, nodes wanting to receive multicast

packets), for packets with the IPv6 NS Solicited-node Multicast Address and continue forwarding these NS requests on the link.

— MLD protocol offload is supported for either MLD Multicast Listener Query packets or MLD Multicast Address and Source Specific Query packets that check for IPv6 multicast listeners with the Solicited-node Multicast Address placed in the IPv6 destination address field of the IPv6 NS packets that are off-loaded by Foxville.

— IPv6 MLD queries, with the Solicited-node Multicast Address placed in the IPv6 destination address field of the IPv6 NS packets that are off-loaded by Foxville (as defined in RFC 2710 and RFC 3810). The MLDv2 Multicast Listener Report messages returned by firmware to MLDv2 Multicast Listener Query messages which concern the device, contain a Multicast Address Record for each configured Solicited IPv6 addresses (up to 2). Other fields are returned as follows:

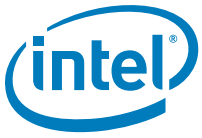
- Number of Sources = 0 (no Source Address fields supplied)
- Record Type = 2 (MODE_IS_EXCLUDE)
- Aux Data Len = 0 (no Auxiliary Data fields supplied)

4. mDNS proxy offload
   — Multicast DNS (mDNS) is used to advertise and locate services on the local network. Its proxy offload requires Foxville to respond to mDNS queries as well as keeping the network connectivity of a system while the system is in sleep state and wake the system when a service is requested from the system.
   — For more information on Foxville functionality and enablement for mDNS Proxy Offload. See section 5.8.3
5. In addition to the D3 low power functionality, by setting *D0_PROXY* bit to 1b, Foxville enables these features in D0 and enables the system to be in a low power S0x state for longer durations to increase system power savings.

5.8.1 Protocol Offload Activation in D3

To enable protocol offload, the software device driver should implement the following steps before D3 entry:

1. Read *MANC.MPROXYE* bit to verify that proxying is supported by management.
2. Clear all pending proxy status bits in the Proxying Status (PROXYS) register.
3. Program the Proxying Filter Control (PROXYFC) register to indicate the type of packets that should be forwarded to manageability for proxying and then program the necessary data to the IPv4/v6 Address Table (IP4AT, IP6AT) and the Flexible Host Filter Table (FHFT / FHFT_EXT) registers.
4. Set the *WUFC.FW_RST_WK* bit to 1b to initiate a wake if firmware reset was issued when in D3 state and proxying information was lost.
5. Take ownership of the Management Host interface semaphore (*SW_FW_SYNC.SW_MNG_SM* register bit) using the flow defined in Section 4.8.1 to send Protocol Offload information to Firmware.
6. Read and clear the *FWSTS.FWRI* firmware reset indication bit.
   — If a firmware reset was issued as reported in the *FWSTS.FWRI* bit, the software device driver should clear the bit and then re-initialize the protocol offload list even if firmware keeps the protocol offload list on a move from D3 to D0 (See note in Section 10.6.3.7.2).
7. Verify that the *HICR.En* bit (See Section 8.22.2) is set 1b, which indicates that the shared RAM interface is available.
8. Write proxying information in the shared RAM interface located in addresses 0x8800-0x8EFF using the format defined in Section 10.6.3.7. All addresses should be placed in networking order.
9. Once information is written into the shared RAM software should set the *HICR.C* bit to 1b.



10. Poll the *HICR.C* bit until bit is cleared by firmware indicating that the command was processed and verified that the command completed successfully by checking that the *HICR.SV* bit was set.
11. Read the firmware response from the shared RAM to verify that data was received correctly.
12. Return to 8. if additional commands need to be sent to Firmware.
13. Release management Host interface semaphore (*SW_FW_SYNC.SW_MNG_SM* register bit) using the flow defined in Section 4.8.2.
14. Verify that a firmware reset was not initiated during the proxying configuration process by reading the *FWSTS.FWRI* firmware reset indication bit. If a firmware reset was initiated. Return to step 1.
15. Set *WUC.PPROXYE* bit to 1b and enable entry into D3 low power state.
16. Once Foxville moves back into D0 state, the software device driver needs to clear the *WUC.PPROXYE* bit, PROXYS, and PROXYFC registers until the next time the system moves to a low power state with proxying enabled.

Normally, after enabling wake-up or proxying, system software moves the device to D3 low power state by writing a 11b to the PCI *PMCSR.Power State* field.

Once proxying is enabled by setting the *WUC.PPROXYE* bit to 1b and device is placed in the D3 low power state, Foxville monitors incoming packets, first filtering them according to its standard address filtering method, then filtering them with all of the proxying filters enabled in the PROXYFC register. If a packet passes both the standard address filtering and at least one of the enabled proxying filters and does not pass any of the enabled wake-up filters, Foxville:

1. Executes the relevant protocol offload for the packet and not forward the packet to the host.
2. Set one or more bits in the Proxying Status (PROXYS) register according to the proxying filters matched.

Note: Foxville sets more than one bit in the PROXYS register if a packet matches more than one filter.

3. Wakes the system and forwards a packet that matches the proxying filters but can't be supported by the host for further processing if configured to do so by the software device driver via the Set Firmware Proxying Configuration command using the shared RAM interface (See Section 10.6.3.7.2).

Notes:

1. When the device is in D3, a packet that matches both one of the enabled proxying filters as defined in the PROXYFC register and one of the enabled wake-up filters as defined in the WUFC register only wakes up the system and protocol offload (proxying) does not occur.
2. Protocol offload is not executed for illegal packets with CRC errors or checksum errors and the packets are silently discarded.
3. Once a packet that meets the criteria for proxying is received, Foxville should respond to the request after less than 60 Seconds.

5.8.2 Protocol Offload Activation in D0

To enable protocol offload in D0,the software device driver should implement the following steps:

1. Read *MANC.MPROXYE* bit to verify that proxying is supported by management.
2. Clear all pending proxy status bits in the Proxying Status (PROXYS) register.
3. Program the Proxying Filter Control (PROXYFC) register to indicate the type of packets that should be forwarded to manageability for proxying and then program the necessary data to the IPv4/v6 Address Table (IP4AT, IP6AT) and the Flexible Host Filter Table (FHFT / FHFT_EXT) registers.

4. Take ownership of the management host interface semaphore (*SW_FW_SYNC.SW_MNG_SM* register bit) using the flow defined in Section 4.8.1 to send protocol offload information to firmware.
5. Verify that the *HICR.En* bit (See Section 8.22.2) is set 1b, which indicates that the shared RAM interface is available.
6. Read and clear the *FWSTS.FWRI* firmware reset indication bit.
   — If a firmware reset was issued as reported in the *FWSTS.FWRI* bit, the software device driver should clear the bit and then re-initialize the protocol offload list.
7. Write proxying information in the shared RAM interface located in addresses 0x8800-0x8EFF using the format defined in Section 10.6.3.7. All addresses should be placed in networking order.
8. Once information is written into the shared RAM, software should set the *HICR.C* bit to 1b.
9. Poll the *HICR.C* bit until the bit is cleared by firmware indicating that command was processed and verified that the command completed successfully by checking that the *HICR.SV* bit was set.
10. Read the firmware response from the shared RAM to verify that data was received correctly.
11. Return to step 7. if additional commands need to be sent to firmware.
12. Release the management host interface semaphore (*SW_FW_SYNC.SW_MNG_SM* register bit) using the flow defined in Section 4.8.2.
13. Verify that a firmware reset was not initiated during the proxying configuration process by reading the *FWSTS.FWRI* firmware reset indication bit. If a firmware reset was initiated, return to step 1.
14. Set the *PROXYFC.D0_PROXY* bit to 1b.
15. Set the *WUC.PPROXYE* bit to 1b to enable protocol offload.

Once proxying is enabled in D0 by setting both the *WUC.PPROXYE* bit to 1b and the *PROXYFC.D0_PROXY* bit to 1b, Foxville monitors incoming packets, first filtering them according to the standard address filtering method and then filtering them according to the proxying filters enabled in the PROXYFC register. If a packet passes both the standard address filtering and at least one of the enabled proxying filters then Foxville:

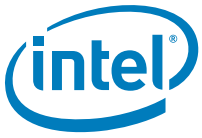
1. Executes the relevant protocol offload for the packet and not forward the packet to the host.
2. Set one or more bits in the Proxying Status (PROXYS) register according to the proxying filter that detected a match.

Note: Foxville sets more than one bit in the PROXYS register if a packet matches more than one filter.

3. Discard silently illegal packets with CRC errors or checksum errors without implementing the protocol offload.
4. Forward a packet that matches the proxying filters but can't be supported by firmware to the host for further processing, if configured to do so by the software device driver via the Set Firmware Proxying Configuration command using the shared RAM interface.

5.8.3 mDNS Proxy Offload

Foxville uses multicast DNS (mDNS) to advertise and locate services on the local network. The mDNS responder system component holds a database of registered services. When the system is in S0 state, the mDNS responder and related system components are the sole entities responsible for auto-configuring the LAN interface, sending service announcements, and handling service query processing. In contrast, the offload component (the mDNS proxy) causes services (shared printers, iTunes libraries etc.) to continue to be discovered (and virtually available) when the system is in a low power state.



The mDNS proxy is activated on demand, through a request from the main mDNS responder. This activity is triggered by the system PM module making the decision to enter a low power state in a system that supports mDNS proxy. Prior to the Sx entry, the internal mDNS record database is sent to the mDNS proxy. This configuration is expected to be used by the proxy to respond to queries and wake the system when a service access is detected. Waking the host causes the proxy function to be disabled.

The mDNS proxy architecture is based on the receive filter and the management controller, the filter is responsible to parse and filter incoming packets and pass the relevant packets to the management controller or wake the system if a packet matches one of the wake up filters. The host driver is responsible to properly configure the filter. The host driver is also responsible to configure the management controller using the Set mDNS Proxy Command for proper operation.

The configuration for proxy includes:

- A list of IPv4 and IPv6 addresses for the interface and enablement of their protocol offload (see Section 5.8.1).
  - — Foxville supports mDNS proxy of up to 1 IPv4 addresses and/or up to 2 IPv6 addresses.
- An array of DNS Resource Records (RRs) to be proxyied by firmware
- An array of UDP and TCP port numbers for the services

The configuration is loaded to the Flash. Refer to Figure 3-8, Figure 3-9, and to Section 6.1.1.36, 6.1.1.37. In order to prevent Flash wear out, the host driver writes these Flash areas only if, since the last time the system went into a sleep state, a record has been modified/added, or if the FW image has been updated. Then, prior to entering a sleep state, the host uses the Set mDNS Proxy command defined in Section 10.6.3.7.5 to activate the mDNS proxy.

When activated, the mDNS proxy must act as a responsible mDNS responder as described in the multicast DNS draft RFC http://files.multicastdns.org/draft-cheshire-dnsext-multicastdns.txt, the DNS based Service Discovery http://files.dns-sd.org/draft-cheshire-dnsext-dns-sd.txt and the mDNS Offload - Draft 1.0 document.

It needs to:

- Listen for both unicast and multicast DNS queries on UDP port 5353
- Respond with a unicast or multicast answer depending on the QU/QM flag
- Not respond if the answer it would give is already in the answer section and the RR TTL is over half the original TTL
- Properly handle queries that span multiple packets (truncated bit is set)
- Support negative responses for known-missing rrtype “A” and “AAAA” queries
- Implement the random delays before responding to non-probe queries, as required to avoid packet storms
- Support merging answers from multiple queries into a single response
- Support legacy DNS queries
- Respond to ARP and IPv6 neighbor solicitation requests
- Respond ICMP PING requests
- Wake the system if one of the offload services is requested or a pre-defined wake up/Magic packet was received. The service wake detection wake up is configured by the host driver using the WFUTPF[31:0], RFUTPF, RWPFC registers.
- Wake the system if the link was lost and re-gained while sleeping
- Wake the system if mDNS name conflict was detected

- Provide the wakeup-reason to the software device driver that details why the system is being woken up.

Proposed configuration of the receive and wakeup filters:

Table 5-10. mDNS Offload Configuration

Frame Type	Address/Protocol	Why Needed	Foxville Implementation Filter
ARP Request	Local IPv4 address/ARP	Maintain IPv4 connectivity	PROXYFC.ARP / PROXYFC.ARP_Directed
IGMPv2	224.0.0.251/IGMP	Maintain presence in mDNS group	PROXYFCEX.IGMP / PROXYFCEX.IGMP_mDirected
Multicast mDNS	224.0.0.251/UDP/5353 FF02::FB/UDP/5353	Listen to multicast mDNS queries and respond when proper	PROXYFCEX.mDNS / PROXYFCEX.mDNS_mDirected
Unicast mDNS	Local IPv4 address/UDP/5353 Local IPv6 address/UDP/5353	Listen to unicast mDNS queries and respond when proper	PROXYFCEX.mDNS / PROXYFCEX.mDNS_uDirected
ICMP	Local IPv4 address/ICMPv4 Local IPv6 address/ICMPv6	PING support	PROXYFCEX.ICMPv4 / PROXYFCEX.ICMPv4_uDirected PROXYFCEX.ICMPv6 / PROXYFCEX.ICMPv6_uDirected
NS/MLD	Local IPv6 address/NS ff02::1/MLD	Maintain IPv6 connectivity	PROXYFC.NS / PROXYFC.NS_Directed
mDNS Proxy Wake Frame	UDP port TCP Port/SYN	Wake the system when one of the offloaded services is requested	WFUTPF[i].Port/ WFUTPF[i].Port_Control RWPFC
mDNS Proxy Special Wake	Non IPSEC keep alive to UDP 4500 TCP SSH data - port 22 UDP 3283 WU packet	Special WU reasons	RWPFC.NonIPsecKA RWPFC.TCP_SSH_Data RWPFC.MagicUDP
Magic Packet WoL	Magic WoL		Part of APM/ACPI WoL

The host driver is responsible to properly configure the receive filters for mDNS proxy and mDNS wake on LAN. Setting bits in the PROXYFCEX register to enable filters that redirect packets to the management controller indicates mDNS proxy offload is required.

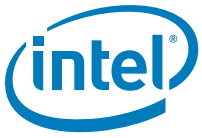
The host driver is also responsible to write the mDNS Records into the Flash area provisioned for it (see Section 6.1.1.36 and Section 6.1.1.37). Refer to the mDNS Proxy SAS document for the exact structure of the mDNS data section to be stored in the Flash.

Note: IP fragments are not supported for mDNS proxy offload, filtering of higher layers (ICMP, TCP/ UDP ports etc.) is not supported on IP fragments.

Note: The mDNS proxy offload will ignore any IPv4 options and silently drop all IPv6 packets with extensions.

5.9 Latency Tolerance Reporting (LTR)

Foxville generates PCIe LTR messages to report service latency requirements for memory reads and writes to the Root Complex (RC) for system power management.



Foxville reports either minimum latency tolerance, maximum latency tolerance or no latency tolerance requirements as a function of link, LAN port and function status. Minimum and maximum latency tolerance values are programmed in the LTRMINV and LTRMAXV registers, respectively by the software device driver to optimize power consumption without incurring packet loss due to receive buffer overflow.

5.9.1 LTR Algorithm

Foxville sends LTR messages according to the following algorithm when the capability is enabled in the LTR capability structure in PCIe configuration space:

1. When link disconnected or port is disabled (transmit and receive activity not enabled) and the *LTRC.LNKDLS_EN* and *LTRC.PDLS_EN* bits are set. respectively, Foxville sends a LTR PCIe message with LTR requirement bits cleared, to indicate that no latency tolerance requirements exists.
2. If Foxville reported following PCIe link-up latency tolerance requirements with any requirement bit set in the PCIe LTR message and the function is placed in D3 low power state via the PMCSR register, Foxville sends a new LTR message with all the requirement bits clear.
3. If Foxville reported following PCIe link-up latency tolerance requirements with any requirement bit set and the *LTR Mechanism Enable* bit in the PCIe configuration space is cleared, Foxville sends a new LTR message with all the requirement bits clear.
4. Foxville sends a LTR message with the value placed in the LTRMAXV register when either one of following conditions exist:
   a. Software set the *LTRC.LTR_MAX* register bit.
   b. Rx EEE LPI state is detected on the Ethernet link and *LTRC.EEEMS_EN* is set (see Section 3.6.6.4).
5. Otherwise, Foxville sends a LTR message with a minimum value.

Note: In all cases, the maximum LTR value sent by Foxville does not exceed the maximum latency values in the Max No-Snoop Latency and Max Snoop Latency Registers in the LTR capability structure of function 0.

Figure 5-4 shows Foxville LTR message generation flow.

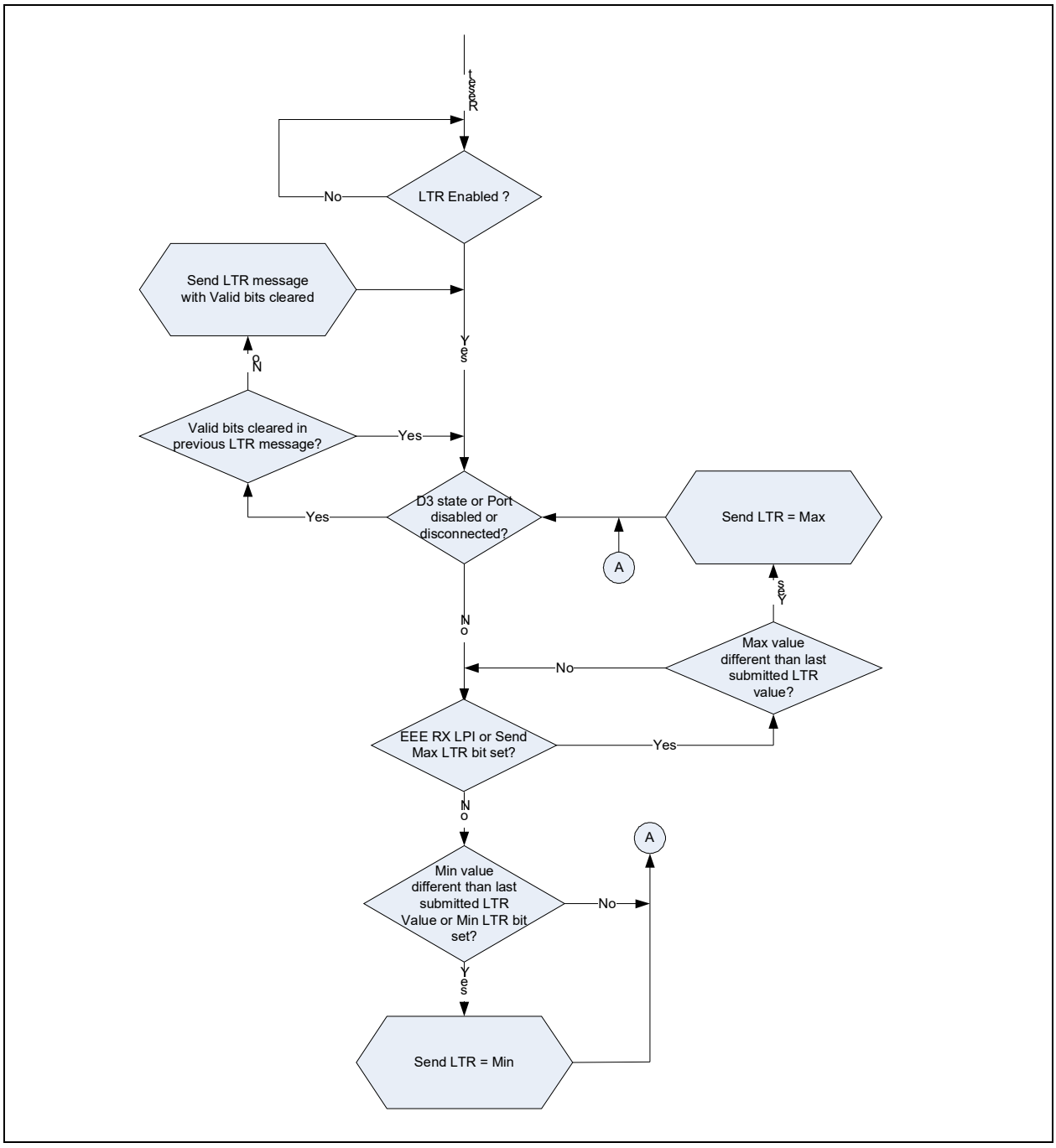
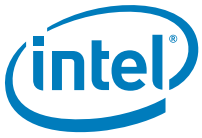


Figure 5-4. PCIe LTR Message Generation Flow



5.9.2 Latency Tolerance Reporting

The PCIe function can request to generate a minimum value LTR, a maximum value LTR, and a LTR message with the requirement bits cleared. Foxville incorporates latency requirements from the function, provided it has LTR messaging enabled and sends a single LTR message in the following manner:

- The acceptable latency values for the message sent upstream by Foxville must reflect the lowest latency tolerance values associated with the function.
  - If the function has no Latency requirement for a certain type of traffic (snoop/non-snoop), the message sent by Foxville does not have the requirement bit corresponding to that type of traffic set.
- Foxville transmits a new LTR message upstream when the capability is enabled and when the function changes the values it has reported internally in such a way as to change the incorporated value reported previously by Foxville.

The PCIe function in Foxville reports support of LTR messaging in the configuration space by:

- Setting the *LTR Mechanism Supported* bit in the PCIe Device Capabilities 2 configuration register (support defined by *LTR_EN* bit in Initialization Control Word 1 NVM word, that controls enabling of the LTR structures).
- Supporting the LTR capability structure in the PCIe configuration space.

To enable generating LTR messages, the *LTR Mechanism Enable* bit in the Device Control 2 configuration register of function 0 should be set.

Note: If the function does not have LTR messaging enabled, it is considered a function that does not have any latency tolerance requirements.

5.9.2.1 Conditions for Generating LTR Message with the Requirement Bits Cleared

When LTR messaging is enabled, Foxville's function sends a LTR message with the requirement bits cleared in the following cases:

1. Following PE_RST_N assertion (PCIe reset) after LTR capability is enabled.
2. LAN port is disabled (both *RCTL.RXEN* and *TCTL.EN* are cleared), receive buffer is empty and *LTRC.PDLS_EN* is set.
3. LAN port is disconnected, BMC to Host traffic is disabled (*MANC.EN_BMC2HOST* = 0) and *LTRC.LNKDLS_EN* is set.
4. Function is not in D0a state.
5. When the LSNP and LNSNP bits are cleared in the LTRMINV register and minimum LTR value needs to be sent.
6. When the LSNP and LNSNP bits are cleared in the LTRMAXV register and maximum LTR value needs to be sent.
7. When the *LTR Mechanism Enable* bit in the Device Control 2 configuration register of function 0 was cleared and Foxville sent previously a LTR message with requirement bits set.

When one of the previous conditions exist, Foxville sends a LTR message with the requirement bits cleared.

Note: If the PCIe function is disabled, it does not generate latency tolerance requirements.

5.9.2.2 Conditions for Generating LTR Message with Maximum LTR Value

When LTR messaging is enabled and conditions to send a LTR message with valid bits cleared do not exist, Foxville functions send a maximum value LTR message, with the values programmed in the LTRMAXV register in the following cases:

1. Following a software write of 1b to the *LTRC.LTR_MAX* bit and the last PCIe LTR message sent had a latency tolerance value different then the value specified in the LTRMAX register.
2. Rx EEE LPI state is detected on the Ethernet link and *LTRC.EEEMS_EN* is set.
3. When updated data was written to the LTRMAXV register and conditions defined in step 1. or step 2. to send a LTR message with a maximum value exists.

When one of the previous conditions exist and the function is enabled and conditions to send a LTR message with requirement bits cleared (See Section 5.9.2.1) doesn't exist, Foxville sends a LTR message with the values programmed in the LTRMAXV register.

Note: When the *LTRC.LTR_MAX* bit is cleared, Foxville sends a LTR message with the value placed in the LTRMINV register, if the value is smaller than the value placed in the LTRMAXV register.

5.9.2.3 Conditions for Generating LTR Message with Minimum LTR Value

When LTR messaging is enabled, Foxville's function sends a minimum value LTR message, with the values programmed in the LTRMINV register in the following cases:

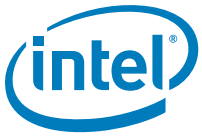
1. Following a software write of 1b to the *LTRC.LTR_MIN* bit and the last PCIe LTR message sent had a latency tolerance value different then the value specified in the LTRMINV register.
2. When updated data was written to the LTRMINV register and conditions to send a LTR message with the requirement bits cleared (see Section 5.9.2.1) or maximum value LTR (see Section 5.9.2.2) do not exist.

Note: If a LTR message that indicates that best possible service is requested needs to be sent, the latency tolerance value in the LTRMINV and LTRMAXV registers should be programmed to 0x0 with the appropriate requirement bits set. In this case, Foxville sends a LTR message with both the value and scale fields cleared to zeros.

5.9.2.4 Recommended Programming of the LTR Parameters

The LTR message on the PCIe inform the requested latency on which the PCIe must become active to the downstream PCIe port of the system. To be eligible to enter L1.2 PCIe sub-state, the Threshold Value/Scale in the L1 PM Substates Control 1 Register should be lower than the LTRV/Scale parameters in the last LTR message generated by Foxville. This section provides recommended programming values for the following LTR parameters:

LTRV and Scale parameters in the **LTRMINV** register: The LTRV and Scale parameters in the LTRMINV register are used to inform the requested PCIe latency becoming active when the in the D0a state and the link is active (and not in the EEE state). The LTR value should be programmed to the receive packet buffer size divided by the link rate minus 1 usec (for some relaxation). So, in case that a single receive packet buffer (RPB) of 34KB is used then at a link rate of 1Gb/s the value and scale should be programmed to ~277 usec (278.5 minus 1). At 2.5Gb/s it would be ~110 usec (111.4 minus 1) and so on. When using 2 x RPBs then the user should program the LTR value according to the RPB on which possible long enough burst (longer burst traffic than the size of the RPB) is expected. For example: assume 2 x RPBs



while the smaller one is 8KB and the larger one is 26KB. Assume also that possible long enough burst is expected only on the larger RPB then the LTR value and scale should be programmed to ~211 usec (212.9 minus 1) at 1Gb/s.

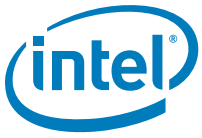
LTRV and Scale parameters in the **LTRMAXV** register: The LTRV and Scale parameters in the LTRMAXV register are used to inform the requested PCIe latency becoming active in non D0a or when the link is inactive or when the link is in the EEE state. A value of ~300 usec should work for most cases at 1Gb/s link with a single RPB on which bursty traffic is not expected immediately after exiting the EEE state. If one would want to avoid any chance of lost packets overcoming also a possible long enough burst immediately after exiting the EEE state then the same values programmed to the LTRMINV register should be copied to the LTRMAXV register as well.

Threshold Value and **Scale** in the **L1 PM Substates Control 1** Register: The threshold value and scale should be programmed to the latency of Foxville exiting the L1.2 state back to L0 state plus some margin. It should cover the time reported by both the T_POWER_ON and the T-Common-mode parameters in the L1 PM Sub states Capabilities Register plus some margin. So, recommended value that should be programmed to the Threshold is ~75 usec.

6.0 Non-Volatile Memory Map

6.1 NVM General Summary Table

Word Address	Used By	Word Name	Reference
0x0000	HW	Ethernet Individual Address 0	Section 6.1.1.1
0x0001	HW	Ethernet Individual Address 1	Section 6.1.1.2
0x0002	HW	Ethernet Individual Address 2	Section 6.1.1.3
0x0003	SW	Compatibility Bytes	Section 6.1.1.4
0x0004	SW	Reserved	Section 6.1.1.5
0x0005	SW	Dev_starter Version	Section 6.1.1.6
0x0006	SW	OEM configuration 1	Section 6.1.1.7
0x0007	SW	OEM configuration 2	Section 6.1.1.8
0x0008	HW	PBA Number 0	Section 6.1.1.9
0x0009	HW	PBA Number 1	Section 6.1.1.10
0x000A	HW	Initialization Control Word 1	Section 6.1.1.11
0x000B	HW	Subsystem ID	Section 6.1.1.12
0x000C	HW	Subsystem Vendor ID	Section 6.1.1.13
0x000D	HW	Device ID	Section 6.1.1.14
0x000E	HW	Vendor ID	Section 6.1.1.15
0x000F	HW	Initialization Control Word 2	Section 6.1.1.16
0x0010	FW	Firmware Image Pointer	Section 6.1.1.17
0x0011	HW	Flash Device Size	Section 6.1.1.18
0x0012	HW	EEPROM Sizing and Protected Fields	Section 6.1.1.19
0x0013	HW	Initialization Control Word 4	Section 6.1.1.20
0x0014	HW	PCIe L1 Exit latencies	Section 6.1.1.21
0x0015	HW	PCIe Completion Timeout Configuration	Section 6.1.1.22
0x0016	HW	MSI-X Configuration	Section 6.1.1.23
0x0018	HW	PCIe Init Configuration 1	Section 6.1.1.24
0x0019	HW	PCIe Init Configuration 2	Section 6.1.1.25
0x001A	HW	PCIe Init Configuration 3	Section 6.1.1.26
0x001B	SW	PCIe Control 1	Section 6.1.1.27
0x001C	HW	LED 1,3 Configuration Defaults	Section 6.1.1.28
0x001D	HW	Dummy Device ID	Section 6.1.1.29
0x001E	HW	Device Rev ID	Section 6.1.1.30
0x001F	HW	LED 0,2 Configuration Defaults	Section 6.1.1.31
0x0020	HW	Software Defined Pins Control	Section 6.1.1.32



Word Address	Used By	Word Name	Reference
0x0021	HW	Functions Control	Section 6.1.1.33
0x0022	HW	LAN Power Consumption	Section 6.1.1.34
0x0024	HW	Initialization Control Word 3	Section 6.1.1.35
0x0025	HW	mDNS Records Area Offset	Section 6.1.1.36
0x0026	HW	mDNS Records Area Size	Section 6.1.1.37
0x0027	HW	CSR Auto Configuration Power-Up Pointer	Section 6.1.1.38
0x0028	HW	PCIe Control 2	Section 6.1.1.39
0x0029	HW	PCIe Control 3	Section 6.1.1.40
0x002A	HW	CDQM Memory Base Low	Section 6.1.1.41
0x002B	HW	CDQM Memory Base High	Section 6.1.1.42
0x002C	HW	End of RO Area pointer (protected area)	Section 6.1.1.43
0x002D	HW	Start of RO Area pointer (protected area)	Section 6.1.1.44
0x002E	HW	Watchdog Configuration	Section 6.1.1.45
0x002F	HW	VPD Pointer	Section 6.1.1.46
0x0030	HW	Setup Options PCI Function	Section 6.1.1.47
0x0031	HW	Configuration Customization Options PCI Function	Section 6.1.1.48
0x0032	HW	PXE Version	Section 6.1.1.49
0x0033	HW	IBA Capabilities	Section 6.1.1.50
0x0034	HW	PCIe PHY Configuration 0 Low	Section 6.1.1.51
0x0035	HW	PCIe PHY Configuration 0 High	Section 6.1.1.52
0x0036	HW	iSCSI Option ROM Version	Section 6.1.1.53
0x0037	HW	Alternate MAC Address Location	Section 6.1.1.54
0x0038	HW	PCIe PHY Configuration 1 Low	Section 6.1.1.55
0x0039	HW	PCIe PHY Configuration 1 High	Section 6.1.1.56
0x003A	HW	PCIe PHY Configuration 2 Low / Reset to PCIe PHY Delay (x40us)	Section 6.1.1.57
0x003B	HW	Reserved	Section 6.1.1.58
0x003C	FW	PXE VLAN pointer	Section 6.1.1.59
0x003D	HW	iSCSI Boot Configuration Pointer	Section 6.1.1.60
0x003E	HW	Reserved	Section 6.1.1.61
0x003F	SW	Checksum Word	Section 6.1.1.62
0x0040	SW	Free Provisioning Area Pointer	Section 6.1.1.63
0x0041	SW	Free Provisioning Area Size	Section 6.1.1.64
0x0042	SW	Image Unique ID 0	Section 6.1.1.65
0x0043	SW	Image Unique ID 1	Section 6.1.1.66
0x0044	SW	PCIe L1 Substates Capability Low	Section 6.1.1.67
0x0045	SW	PCIe L1 Substates Capability High	Section 6.1.1.68
0x0046	SW	PCIe L1 Substates Control 1st Low	Section 6.1.1.69
0x0047	SW	PCIe L1 Substates Control 1st High	Section 6.1.1.70
0x0048	SW	PCIe L1 Substates Control 2nd (only lower byte taken)	Section 6.1.1.71
0x0049	SW	PTM Setting	Section 6.1.1.72
0x004A	HW	EXP. ROM Boot Code Section Pointer	Section 6.1.1.73

Word Address	Used By	Word Name	Reference
0x004B	SW	ULP Capability Enable	Section 6.1.1.74
0x004C	SW	Reserved	Section 6.1.1.75
0x004D	FW	Reserved	Section 6.1.1.76
0x004E	SW	Reserved	
0x004F	SW	Reserved	
0x0050	FW	RO Commands Version	Section 6.1.1.77
0x0051	FW	Firmware Module Configuration Pointer	Section 6.1.1.78
0x0055	HW	PCIe PHY Configuration 4 High	Section 6.1.1.82
0x0056	HW	PCIe PHY Configuration 5 Low	Section 6.1.1.83
0x0057	HW	PCIe PHY Configuration 5 High	Section 6.1.1.84
0x0058 ... 0x007F	FW	Reserved	

6.1.1 Common and Lan Port 0 Section Summary Table

Start of Shadow RAM - Sector 0 and Sector 1 at addresses 0x0000 and 0x0800.

For inner structure please See section 6.1

6.1.1.1 Ethernet Individual Address 0 - 0x0000

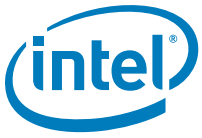
Bits	Field Name	Default NVM Value	Description
15:0	Ethernet Individual Address 0	0x000000c9a000	The Ethernet Individual Address (IA) is a 6-byte field that must be unique for each NIC, and thus unique for each copy of the EEPROM image. The first three bytes are vendor specific. For example, the IA is equal to [00 AA 00] or [00 A0 C9] for Intel products. The value from this field is loaded into the Receive Address Register 0 (RAL0/RAH0).

6.1.1.2 Ethernet Individual Address 1 - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Ethernet Individual Address 1		The second word of the Ethernet Individual Address

6.1.1.3 Ethernet Individual Address 2 - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Ethernet Individual Address 2		The third (and last) word of the Ethernet Individual Address



6.1.1.4 Compatibility Bytes - 0x0003

Bits	Field Name	Default NVM Value	Description
15:13	Reserved	0x0	Reserved
12	IT (Industrial Temperature)	0x0	Valid values are: 0x0 Disabled 0x1 Enabled
11	NIC/LOM	0x1	NIC/LOM (0=NIC; 1=LOM). Valid values are: 0x0 NIC 0x1 LOM
10	Server card	0x1	Server card (0=Client; 1=Server). Valid values are: 0x0 Client 0x1 Server
9	Client card	0x0	Client card (0=Server; 1=Client). Valid values are: 0x0 Server 0x1 Client
8	Retail/OEM card	0x1	Retail/OEM card (0=Retail; 1=OEM). Valid values are: 0x0 Retail 0x1 OEM
7:6	Reserved	0x0	Reserved
5	Reserved	0x1	Reserved
4	Reserved	0x0	SMBus connected (0=Not Connected; 1=Connected). Valid values are: 0x0 Not Connected 0x1 Connected
3	Reserved	0x0	Reserved
2	PCI bridge NOT present	0x0	PCI bridge NOT present. Valid values are: 0x0 PCI bridge NOT present 0x1 PCI bridge present
1:0	Reserved	0x0	Reserved

6.1.1.5 Reserved - 0x0004

Bits	Field Name	Default NVM Value	Description
15:12	Reserved	0xF	
11:8	Reserved	0xF	
7:4	Reserved	0xF	
3:0	Reserved	0xF	

6.1.1.6 Dev_starter Version - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Map Version		

6.1.1.7 OEM configuration 1 - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	OEM configuration	0xFFFF	

6.1.1.8 OEM configuration 2 - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	OEM configuration	0xFFFF	

6.1.1.9 PBA Number 0 - 0x0008

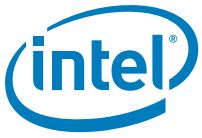
Bits	Field Name	Default NVM Value	Description
15:0	PBA 0	0xFFFF	The first word of the Printed Board Assembly (PBA). The PBA is a nine-digit number used for Intel manufactured NICs that are stored in a four-byte field. The dash itself is not stored; neither is the first digit of the 3-digit suffix, as it is always zero. Note that through the course of hardware ECOs, the suffix field (byte 4) increments. The purpose of this information is to allow customer support or any user to identify the exact revision of the product. Note that this PBA number is unrelated to the MSI-X Pending Bit Array (PBA). If this word equals to 0xFAFA then the second word is a pointer to the legacy "PBA section" structure in the NVM.

6.1.1.10 PBA Number 1 - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	PBA 1	0xFFFF	The second word of the Printed Board Assembly (PBA).

6.1.1.11 Initialization Control Word 1 - 0x000A

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	
14	Reserved	0x1	Reserved
13	LTR_EN	0x1	LTR capabilities reporting enable. 0 - Do not report LTR support in the PCIe configuration Device Capabilities 2 register. 1 - Report LTR support in the PCIe configuration Device Capabilities 2 register. Defines default setting of LTR capabilities reporting (See Section 9.4.5.11). Valid values are: 0x0 - Do not report LTR support in the PCIe configuration space. 0x1 - Report LTR support in the PCIe configuration Device Capabilities 2
12	VPD_EN	0x1	
11:7	reserved	0x0	
6	Reserved	0x0	Reserved.



Bits	Field Name	Default NVM Value	Description
5	Deadlock Timeout Enable	0x1	If set, a device granted access to the EEPROM or Flash that does not toggle the interface for more than 2 seconds will have the grant revoked. See Section 3.3.7
4	LAN PLL Shutdown Enable	0x0	Reserved
3	Reserved	0x1	Reserved
2	DMA clock gating	0x1	When set Disables DMA clock gating power saving mode.
1	Load Subsystem IDs	0x1	When this bit is set to 1b the device loads its PCIe Subsystem ID and Subsystem Vendor ID from the EEPROM (Subsystem ID and Subsystem Vendor ID EEPROM words).
0	Reserved	0x1	Reserved

6.1.1.12 Subsystem ID - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	Subsystem ID	0x0000	If the Load Subsystem IDs in Initialization Control Word 1 EEPROM word is set, the Subsystem ID word in the Common section is read in to initialize the PCIe Subsystem ID. Default value is 0x0 (See Section 9.4.14).

6.1.1.13 Subsystem Vendor ID - 0x000C

Bits	Field Name	Default NVM Value	Description
15:0	Subsystem Vendor ID	0x8086	If the Load Subsystem IDs bit in Initialization Control Word 1 is set, the Subsystem Vendor ID word is loaded to the PCIe Subsystem Vendor ID field in the PCIe configuration space. The default value is 0x8086.

6.1.1.14 Device ID - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	Device ID	0xFFFF	The device ID in the PCIe configuration space is dictated by the device SKU. See Section 1.5.6 for device IDs table per SKU.

6.1.1.15 Vendor ID - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	Vendor ID	0x8086	If the Load Vendor/Device IDs bit in Initialization Control Word 1 EEPROM word is set, this word is initializes the PCIe Vendor ID. The default value is 0x8086 (See Section 9.4.1). Note: If a value of 0xFFFF is placed in the Vendor ID EEPROM word, the value in the PCIe Vendor ID register will return to the default 0x8086 value.

6.1.1.16 Initialization Control Word 2 - 0x000F

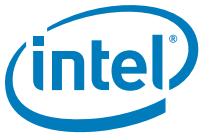
Bits	Field Name	Default NVM Value	Description
15	APM PME# Enable	0x1	Initial value of the Assert PME On APM Wakeup bit in the Wake Up Control (WUC.APMPME) register supporting G3 to S5 wakeup. See Section 8.20.1
14	Reserved	0x0	
13:12	Reserved	0x0	
11	Reserved	0x0	
10	FRCSPD	0x0	Default setting for the Force Speed bit in the Device Control register (CTRL[11]). See Section 8.2.1
9	FD	0x1	Default setting for duplex setting. Mapped to CTRL[0]. See Section 8.2.1
8	TX_LPI_EN	0x1	Enable entry into EEE LPI on TX path. See Section 8.24.5 0b - Disable entry into EEE LPI on TX path. 1b - Enable entry into EEE LPI on TX path.
7	Reserved	0x0	Reserved Zero.
6	PHY Power Down Ena	0x0	When set, the internal PHY enters a power down state in the Dx power state if the link is not required. This bit is loaded to CTRL_EXT[20].
5	Reserved	0x0	Reserved
4	Reserved	0x0	
3	Reserved	0x0	Reserved zero.
2	EEE_2_5G_AN	0x1	Report EEE 2.5G capability in Auto-negotiation. 0b - Do not report EEE 2.5G capability in Auto-negotiation. 1b - Report EEE 2.5G capability in Auto-negotiation.
1	EEE_1G_AN	0x1	Report EEE 1G capability in Auto-negotiation. 0b - Do not report EEE 1G capability in Auto-negotiation. 1b - Report EEE 1G capability in Auto-negotiation.
0	EEE_100M_AN	0x1	Report EEE 100M capability in Auto-negotiation. 0b - Do not report EEE 100M capability in Auto-negotiation. 1b - Report EEE 100M capability in Auto-negotiation.

6.1.1.17 Firmware Image Pointer - 0x0010

Bits	Field Name	Default NVM Value	Description
15	4K Ptr type mark	0x1	
14:0	Ptr	0xA2	Pointer to the beginning of the FW image defined in 4KB units.. It is a pointer to the CSS header Section. For CSS header inner structure See section 6.1.29

6.1.1.18 Flash Device Size - 0x0011

Bits	Field Name	Default NVM Value	Description
15:12	Reserved	0x0	



Bits	Field Name	Default NVM Value	Description
11:9	Flash Speed	0x4	The Flash Speed controls the frequency of the SPI Flash clock (after reading the first 4KB block that is done at 3.125MHz). The frequencies below are defined in MHz units. 0x0 - 15.625 0x1 - 19.53 0x2 - 26.04 0x3 - 31.25 0x4 - 39.06 0x5 - rsv 0x6 - rsv 0x7 - rsv
8	Flash Defaults from this word	0x0	
7	Fast read support	0x0	
6:5	Reserved	0x0	
4	Unprotect After Reset	0x0	
3	SST Mode	0x0	
2:0	Flash Size	0x5	Indicates Flash size according to the following equation: Size = 64 KB * 2**(Flash Size field). From 64 KB up to 8 MB in powers of 2. The Flash size impacts the requested memory space for the Flash and expansion ROM BARs in PCIe configuration space. Note: When CSR_Size and Flash_size fields in the EEPROM are set to 0, Flash access BAR in the PCI configuration space is disabled Valid values are: 0x4 - 1 MB 0x5 - 2 MB 0x6 - 4 MB 0x7 - 8 MB

6.1.1.19 EEPROM Sizing and Protected Fields - 0x0012

Provides indication on EEPROM size and protection. If the Enable Protection Bit in this word is set and the signature is valid, the software device driver has read but no write access to this word via the EEC and EERD registers; In this case, write access is possible only via an authenticated firmware interface.

Bits	Field Name	Default NVM Value	Description
15:14	Signature	0x1	The Signature field indicates to the device that there is a valid EEPROM present. If the signature field is 01b, EEPROM read is performed, otherwise the other bits in this word are ignored, no further EEPROM read is performed, and default values are used for the configuration space IDs.
13	Reserved	0x0	
12	PI Features Enabled	0x0	This bit controls the Printing and Imaging features embedded in FW code.
11	Reserved	0x0	
10:0	Start of 2nd protected area	0x7F0	This field defined the beginning of the write protected area up and including word 0x7FF. Setting this field to 0x7FF - Means that there is no 2nd secured area in the Shadow RAM.

6.1.1.20 Initialization Control Word 4 - 0x0013

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	Reserved
7	Reserved	0x0	Reserved
6	Reserved	0x0	Reserved
5:1	PHY_ADD	0x0	PHY address. Value loaded to MDICNFG.PHYADD field. See Section 8.2.5.
0	Reserved	0x1	Reserved

6.1.1.21 PCIe L1 Exit latencies - 0x0014

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x1	Reserved
14:12	L1_Act_Acc_Latency	0x6	Loaded to the "Endpoint L1 Acceptable Latency" field in the "Device Capabilities" in the "PCIe configuration registers" at power up.
11:9	L1 G2 Sep exit latency	0x2	L1 exit latency G2S. Loaded to "Link Capabilities" -> "L2 Exit Latency" at PCIe v2.0 (2.5GT/s) system in Separate clock setting.
8:6	L1 G2 Com exit latency	0x4	L1 exit latency G2C. Loaded to "Link Capabilities" -> "L2 Exit Latency" at PCIe v2.0 (2.5GT/s) system in Common clock setting.
5:3	L1 G1 Sep exit latency	0x4	L1 exit latency G1S. Loaded to "Link Capabilities" -> "L1 Exit Latency" at PCIe v2.0 (2.5GT/s) system in Separate clock setting.
2:0	L1 G1 Com exit latency	0x4	L1 exit latency G1C. Loaded to "Link Capabilities" -> "L1 Exit Latency" at PCIe v2.0 (2.5GT/s) system in Common clock setting.

6.1.1.22 PCIe Completion Timeout Configuration - 0x0015

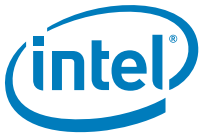
Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	Reserved
7:5	TO_EXT	0x2	Extension to FUNC timeout in memory transactions. Valid values are: 0x0 Timeout: ~4 uSec 0x1 Timeout: ~8 uSec 0x2 Timeout: ~16 uSec 0x3 Timeout: ~32 uSec 0x4 Timeout: ~65 uSec 0x5 Timeout: ~131 uSec 0x6 Timeout: ~262 uSec 0x7 Timeout: ~262 uSec 2
4	Completion Timeout Resend	0x0	When set, enables to resend a request once the completion timeout expired 0b = Do not re-send request on completion timeout. 1b = Re-send request on completion timeout. See section 8.6.1
3:0	Reserved	0x0	Reserved

6.1.1.23 MSI-X Configuration - 0x0016

Bits	Field Name	Default NVM Value	Description
15:11	MSI_X_N	0x4	
10	MSI Mask	0x1	
9:0	Reserved	0x0	

6.1.1.24 PCIe Init Configuration 1 - 0x0018

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	Reserved
14:12	L0s acceptable latency	0x3	Loaded to the "Endpoint L0s Acceptable Latency" field in the "Device Capabilities" in the "PCIe configuration registers" at power up.



Bits	Field Name	Default NVM Value	Description
11:9	L0s G2 Sep exit latency	0x7	L0s exit latency G2S. Loaded to L0s Exit Latency field in the Link Capabilities register in the PCIe configuration registers in PCIe v2.0 (5GT/s) system at Separate clock setting.
8:6	L0s G2 Com exit latency	0x5	L0s exit latency G2C. Loaded to L0s Exit Latency field in the Link Capabilities register in the PCIe configuration registers in PCIe v2.0 (5GT/s) system at Common clock setting.
5:3	L0s G1 Sep exit latency	0x7	L0s exit latency G1S. Loaded to L0s Exit Latency field in the Link Capabilities register in the PCIe configuration registers in PCIe v2.0 (2.5GT/s) system at Separate clock setting.
2:0	L0s G1 Com exit latency	0x5	L0s exit latency G1C. Loaded to L0s Exit Latency field in the Link Capabilities register in the PCIe configuration registers in PCIe v2.0 (2.5GT/s) system at Common clock setting.

6.1.1.25 PCIe Init Configuration 2 - 0x0019

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	Reserved
14	IO_Sup	0x0	I/O Support (effects I/O BAR request) When set to 1b, I/O is supported. When cleared the "I/O Access Enable" bit in the "Command Reg" in the "Mandatory PCI Configuration" area is RO with a value of 0.
13	CSR_conf_en	0x1	Enable CSR access via configuration space. When set enables CSR access via the configuration registers located at configuration address space 0x98 and 0x9C.
12	Serial Number enable	0x1	"Serial number capability" enable. Should be set to one.
11:0	Reserved	0x0	Reserved

6.1.1.26 PCIe Init Configuration 3 - 0x001A

Bits	Field Name	Default NVM Value	Description
15:13	AER Capability Version	0x2	Reserved
12	Cache_Lsize	0x1	Cache Line Size 0b = 64 bytes. 1b = 128 bytes. This bit defines the Cache line size reported in the PCIe mandatory configuration register area. See section 9.3.7
11:10	GIO_Cap	0x2	PCIe Capability Version The value of this field is reflected in the two LSBs of the capability version in the PCIe CAP register (config space offset 0xA2). This field must be set to10b to use extended configuration capability. Note that this is not the PCIe version. It is the PCIe capability version.
9:8	Max Payload Size	0x2	Default PCIe packet size, See section 9.4.5.4 . Valid values are: 0x0 128 Bytes 0x1 256 Bytes 0x2 512 Bytes 0x3 Reserved
7:4	Reserved	0x0	Reserved
3:2	Act_Stat_PM_Sup	0x2	Determines support for active state link power management Loaded into the PCIe Active State Link PM Support register indicating L1 support. See section 9.4.5.7
1	Slot_Clock_Cfg	0x1	When set, the device uses the PCIe reference clock supplied on the connector (for add-in solutions).
0	Reserved	0x0	Reserved

6.1.1.27 PCIe Control 1 - 0x001B

Bits	Field Name	Default NVM Value	Description
13:12	CLTR_INCR	0x0	Consecutive LTR Update Time with increased value. Field defines allowable minimum time interval between consecutive LTR messages with the same or increased value. LTR messages with reduced value are always sent immediately. 00b - No limitation. 01b - 500 uSec 10b - 2000 uSec 11b - 8000 uSec
11	Reserved	0x1	Reserved
10	No_Soft_Reset	0x1	This bit defines Foxville behavior following a transition from the D3hot to D0 Power State. When bit is set no internal reset is issued on transition from D3hot to D0. Value is loaded to the No_Soft_Reset bit in the PMCSR register (See section 9.4.1.4).
9:8	Reserved	0x0	Reserved
7	Debug Enable	0x0	Enable PCIe Configuration write in Debug mode. .
6:0	Reserved	0x0	Reserved

6.1.1.28 LED 1,3 Configuration Defaults - 0x001C

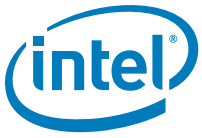
Bits	Field Name	Default NVM Value	Description
15:11	Reserved	0x0	Reserved
10:8	Reserved	0x0	Reserved
7	LED1 Blink	0x0	Initial value of LED1_BLINK field. 0b = Non-blinking. 1b = Blinking. See section 8.2.5 and See section 3.4.3
6	LED1 Invert	0x0	Initial value of LED1_IVRT field. 0b = Active-low output. See section 8.2.5 and See section 3.4.3
5:4	Reserved	0x0	Reserved
3:0	LED1 Mode	0x8	Initial value of the LED1_MODE field specifying what event/state/pattern is displayed on LED1 (ACTIVITY) output. See “Link Mode Encoding” table for all setting options. 0x4 = LINK_2500

6.1.1.29 Dummy Device ID - 0x001D

Bits	Field Name	Default NVM Value	Description
15:0	Dummy Device ID	0x0000	

6.1.1.30 Device Rev ID - 0x001E

Bits	Field Name	Default NVM Value	Description
15	Device Off Enable	1b	Enable Power down when LAN_DISABLE_N pin is asserted or PCIe in Dr state.
14:8	Reserved	0x0	Reserved



Bits	Field Name	Default NVM Value	Description
7:0	DEVREVID	0x0	Device Revision ID The actual device revision ID is the EEPROM value XORed with the hardware default of Rev ID. For step A0 of the device, the default value is zero. See section 9.3.5

6.1.1.31 LED 0,2 Configuration Defaults - 0x001F

Bits	Field Name	Default NVM Value	Description
15	LED2 Blink	0x1	Initial value of LED2_BLINK field. See Section 8.2.5 and See section 3.4.3 0b = Non-blinking. 1b = Blinking.
14	LED2 Invert	0x0	Initial value of LED2_IVRT field. See Section 8.2.5 and See section 3.4.3 0b = Active-low output.
13:12	Reserved	0x0	Reserved
11:8	LED2 Mode	0x4	Initial value of the LED2_MODE field specifying what event/state/pattern is displayed on LED2 (LINK_100) output. See "Link Mode Encoding" table for all setting options. 0x4= LINK_ACTIVITY
7	LED0 Blink	0x0	Initial value of LED0_BLINK field. 0b = Non-blinking. 1b = Blinking.
6	LED0 Invert	0x0	Initial value of LED0_IVRT field. 0b = Active-low output.
5	Global Blink Mode	0x1	Global Blink Mode 0b = Blink at 200 ms on and 200ms off. 1b = Blink at 83 ms on and 83 ms off.
4	Reserved	0x0	Reserved. Set to 0b.
3:0	LED0 Mode	0x7	Initial value of the LED0_MODE field specifying what event/state/pattern is displayed on LED0 (LINK_UP) output. See "Link Mode Encoding" table for all setting options. 0x7= LINK_1000

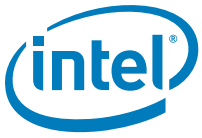
6.1.1.32 Software Defined Pins Control - 0x0020

Bits	Field Name	Default NVM Value	Description
15	SDPDIR[3]	0x0	SDP3 Pin Initial Direction. This bit configures the initial hardware value of the SDP3_IODIR bit in the Extended Device Control (CTRL_EXT) register following power up. See section 8.2.3
14	SDPDIR[2]	0x0	SDP2 Pin Initial Direction. This bit configures the initial hardware value of the SDP2_IODIR bit in the Extended Device Control (CTRL_EXT) register following power up. See section 8.2.3
13	Reserved	0x0	
12	Disable 100 in non-D0a	0x0	Disables 100 Mb/s operation in non-D0a states (See Section 3.5.7.5.4). Sets default value of PHPM. Disable 100 in non-D0a bit.
11	Dis2500	0x0	Disables 2.5Gb/s operation in all power states.
10	Reserved	0x1	Reserved
9	SDPDIR[1]	0x0	SDP1 Pin Initial Direction This bit configures the initial hardware value of the SDP1_IODIR bit in the Device Control (CTRL) register following power up. See section 8.2.1
8	SDPDIR[0]	0x0	SDP0 Pin Initial Direction This bit configures the initial hardware value of the SDP0_IODIR bit in the Device Control (CTRL) register following power up. See section 8.2.1

Bits	Field Name	Default NVM Value	Description
7	SDPVAL[3]	0x0	SDP3 Pin Initial Output Value This bit configures the initial power-on value output on SDP3 (when configured as an output) by configuring the initial hardware value of the SDP3_DATA bit in the Extended Device Control (CTRL_EXT) register after power up. See section 8.2.3
6	SDPVAL[2]	0x0	SDP2 Pin Initial Output Value This bit configures the initial power-on value output on SDP2 (when configured as an output) by configuring the initial hardware value of the SDP2_DATA bit in the Extended Device Control (CTRL_EXT) register after power up. See section 8.2.3
5	WD_SDP0	0x0	When set, SDP[0] is used as a watchdog timeout indication. When reset, it is used as an SDP (as defined in bits 8 and 0). See section 8.2.1
4	Dis1000	0x0	When set, GbE operation and 2.5G are disabled in all power states.
3	Dis1000_nonD0a	0x1	Disables both 2.5 Gb/s and 1000 Mb/s operation in non-D0a states.
2	D3COLD_WAKEUP_ADVEN	0x1	Controls reporting of D3 Cold wake-up support in the Power Management Capabilities (PMC) configuration register. See section 9.4.1.3 When set, D3Cold wake up capability is advertised based on whether AUX_PWR pin is connected to 3.3V to advertise presence of auxiliary power (yes if AUX_PWR is indicated, no otherwise). When 0b, however, D3Cold wake up capability is not advertised even if AUX_PWR presence is indicated. If full 1Gb/sec. operation in D3 state is desired but the system's power requirements in this mode would exceed the D3Cold Wake up-Enabled specification limit (375mA at 3.3V), this bit can be used to prevent the capability from being advertised to the system.
1	SDPVAL[1]	0x0	SDP1 Pin Initial Output Value This bit configures the initial power-on value output on SDP1 (when configured as an output) by configuring the initial hardware value of the SDP1_DATA bit in the Device Control (CTRL) register after power up. See section 8.2.1
0	SDPVAL[0]	0x0	SDP0 Pin Initial Output Value This bit configures the initial power-on value output on SDP0 (when configured as an output) by configuring the initial hardware value of the SDP0_DATA bit in the Device Control (CTRL) register after power up. See section 8.2.1

6.1.1.33 Functions Control - 0x0021

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	Reserved.
14	Reserved	0x0	Reserved.
13:12	Reserved	0x0	Reserved.
11	Reserved	0x0	Reserved
10	BAR32	0x1	When cleared to 0b 64 bit BAR addressing mode is selected. Note: If PREFBAR is set the BAR32 bit should always be 0 (64 bit BAR addressing mode). See section 9.3.11
9	PREFBAR	0x0	0b - BARs are marked as non prefetchable 1b - BARs are marked as prefetchable. Forbided setting option. See section 9.3.11 Notes: 1. Foxville implements Non-prefetchable space in memory BAR, since it has read side effects. This bit is loaded from the PREFBAR bit in the EEPROM. 2. If PREBAR bit is set then the BAR32 bit should be 0b.
8	drop_os2bmc	0x1	0b - Do not drop OS2BMC packets when management buffer is not available. 1b - Drop OS2BMC packets when management buffer is not available. Note: Clearing bit will avoid loss of OS2BMC traffic but may cause head of line blocking on traffic to network.



Bits	Field Name	Default NVM Value	Description
7	DMA Idle Indication	0x1	Lx Coalescing indication is Loaded to PCIEMISC register. Bit indicates when to move out of DMA coalescing either when one of the Foxville ports moves out of DMA coalescing or when PCIe Link moves to L0. 0b - Move out of DMA coalescing when one of Foxville ports stopped DMA coalescing. 1b - Move out of DMA Coalescing when link is in L0 ASPM state and TLPs detected.
6	Lx_decision	0x0	Low power link mode decision is Loaded to PCIEMISC register. 0b - Legacy mode. Decision to move to L0s or L1 based on internal timer. 1b - Decision to move to L0s and L1 based on DMA requirements. Note: .
5	Reserved	0x0	Reserved
4	Reserved	0x0	
3	Reserved	0x0	
2	drop_bmc2os	0x1	
1:0	Reserved	0x0	Defines the slew rate of the NCSI_CLK_OUT, NCSI_CRS_DV and NCSI_RXD pads. If set, the slew rate is high.

6.1.1.34 LAN Power Consumption - 0x0022

Bits	Field Name	Default NVM Value	Description
15:8	LAN D0 Power	0x0	The value in this field is reflected in the PCI Power Management Data Register of the LAN functions for D0 power consumption and dissipation (Data_Select = 0 or 4). Power is defined in 100mW units. The power includes also the external logic required for the LAN function.
7:5	Reserved	0x0	Reserved
4:0	LAN D3 Power	0x0	The value in this field is reflected in the PCI Power Management Data register of the LAN functions for D3 power consumption and dissipation (Data_Select = 3 or 7). Power is defined in 100 mW units. The power also includes the external logic required for the LAN function. The MSBs in the data register that reflects the power values are padded with zeros.

6.1.1.35 Initialization Control Word 3 - 0x0024

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	Reserved
14	Reserved	0x0	Reserved
13	ILOS	0x0	Default setting for the loss-of-signal polarity bit (CTRL[7]). See section 8.2.1
12:11	Interrupt Pin	0x0	Reserved for Interrupt Pin field in Intel devices with more than one port. In this device this field must be set to 0x0 indicating the use of INTA#.
10	APM Enable	0x1	Initial value of Advanced Power Management Wake Up Enable bit in the Wake Up Control (WUC.APME) register. Mapped to CTRL[6] and to WUC[0]. See section 8.2.1 and See section 8.20.1
9	Enable Automatic Crossover	0x1	MDI Flip When set MDI Channel D is exchanged with MDI Channel A and MDI Channel C is exchanged with MDI Channel B.

Bits	Field Name	Default NVM Value	Description
8	ACBYP	0x0	Bypass on-chip AC coupling in RX input buffers 0b -Normal mode; on-chip AC coupling present. 1b - On-chip AC coupling bypassed. Used to set default value of P1GCTRL0.ACBYP
7	LAN Boot Disable	0x1	A value of 1b disables the expansion ROM BAR in the PCI configuration space.
6	PHY Link Down Ena	0x0	1b - The Ethernet link goes down (disconnected) when the device is set to D3 state and the link functionality is not required or device off state. 0b - Autonomous link disconnect at D3 or device off is not enabled. This bit is mapped to CTRL_EXT[24].
5:4	Reserved	0x0	This field is reserved and must be set to 0x0 (it was used to be LINK_MODE in Foxville)
3	Reserved	0x0	
2	External MDIO	0x0	When set PHY management interface is via external MDIO interface. Loaded to MDICNFG.Destination. See section 8.2.6
1	EXT_VLAN	0x0	Sets the default for CTRL_EXT[26] bit. Indicates that additional VLAN is expected in this system. See section 8.2.3
0	Keep_PHY_Link_Up_En	0x0	Enables No PHY Reset when the Baseboard Management Controller (BMC) indicates that the PHY should be kept on. When asserted, this bit prevents the PHY reset signal and the power changes reflected to the PHY according to the MANC.Keep_PHY_Link_Up value (Enables assertion of Veto bit by Management). Note: This EEPROM bit should be set to the same value for all LAN ports.

6.1.1.36 mDNS Records Area Offset - 0x0025

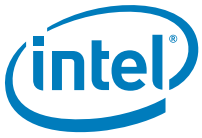
Bits	Field Name	Default NVM Value	Description
15:0	mDNS Records Area Offset	0x805d	

6.1.1.37 mDNS Records Area Size - 0x0026

Bits	Field Name	Default NVM Value	Description
15:0	mDNS Records Area Size	0x0010	

6.1.1.38 CSR Auto Configuration Power-Up Pointer - 0x0027

Bits	Field Name	Default NVM Value	Description
15:11	reserved	0x0	
10:0	CSR Auto Configuration Power-Up Pointer	0x7ff	This field points to LPG Reset CSR Auto Configuration Section. For LPG Reset CSR Auto Configuration inner structure See section 6.1.6



6.1.1.39 PCIe Control 2 - 0x0028

Bits	Field Name	Default NVM Value	Description
15:14	Reserved	0x0	Reserved
13	ECRC Generation for	0x0	
12	ECRC Check	0x1	Loaded into the ECRC Check Capable bit of the PCIe configuration registers 0b - Function is not capable of checking ECRC 1b - Function is capable of checking ECRC
11	ECRC Generation	0x1	Loaded into the ECRC Generation Capable bit of the PCIe configuration registers. 0b - Function is not capable of generating ECRC 1b - Function is capable of generating ECRC
10	FLR capability enable	0x1	FLR capability Enable bit is loaded to the "PCIe Control 2" word in the PCIe configuration space.
9:6	FLR delay	0x1	Delay in microseconds from D0 to D3 move till reset assertion.
5	FLR delay disable	0x1	FLR delay disable. 0 - Add delay to FLR assertion. 1 - Do not add delay to FLR assertion.
4	Reserved	0x0	Reserved
3:1	Flash Size bar	0x4 / 0x5	Indicates Flash size according to the following equation: Size = 64 KB * 2**(Flash Size field). From 64 KB up to 8 MB in powers of 2. The Flash size impacts the requested memory space for the Flash and expansion ROM BARs in PCIe configuration space. Valid values are: 0x4 = 1 MB 0x5 = 2 MB
0	CSR_Size	0x0	The CSR_Size and FLASH_Size fields define the usable FLASH size and CSR mapping window size.

6.1.1.40 PCIe Control 3 - 0x0029

Bits	Field Name	Default NVM Value	Description
15	en_pin_pcie_func_dis	0x0	When set to 1b enables disabling the PCIe function by driving the MCS / PCIE_ENA pin to low during power up or exit from the ULP state (see Section 4.6.3).
14	Reserved	0x0	When set to 1b, SDP_3 pad controls the LAN disable functionality. When the SDP_3 pad is driven low, the LAN port is disabled.
13	nvm_alt_aux_pwr_en	0x0	When set to 1b, SDP_3 pad controls the auxiliary power funstionality. When SDP_3 pad is driven high, it indicates that auxiliary power is provided.
12	Reserved	0x0	Reserved
11	Reserved	0x0	Reserved
10	nvm_aux_pwr_en	0x0	When set to 1b, LAN_DISABLE_N pad controls the auxiliary power functionality. When LAN_DISABLE_N pad is driven high, it indicates that auxiliary power is provided
9	Reserved	0x0	Reserved
8:7	Reserved	0x0	Reserved
6	Reserved	0x0	Reserved
5	Wake_pin_enable	0x1	Enables the use of the wake pin for a PME event in all non L2 power states.
4	DIS Clock Gating in DISABLE	0x1	Disable clock gating when LTSSM is at DISABLE state.

Bits	Field Name	Default NVM Value	Description
3	DIS Clock Gating in L2	0x1	Disable clock gating when LTSSM is at L2 state.
2	DIS Clock Gating in L1	0x1	Disable clock gating when LTSSM is at L1 state
1:0	Reserved	0x0	Reserved

6.1.1.41 CDQM Memory Base Low - 0x002A

Bits	Field Name	Default NVM Value	Description
15:0	CDQML	0x8000	Client Data and Descriptor Queue Memory Low. Loaded to CDQM Register bits 15:0.

6.1.1.42 CDQM Memory Base High - 0x002B

Bits	Field Name	Default NVM Value	Description
15:0	CDQMH	0x0701	Client Data and Descriptor Queue Memory Low. Loaded to CDQM Register bits 31:16.

6.1.1.43 End of RO Area Pointer (protected area) - 0x002C

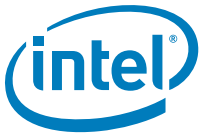
Bits	Field Name	Default NVM Value	Description
15:11	Reserved	0x0	Reserved
10:0	EORO_area	0x0	Defines the end of the area in the EEPROM that is RO. The resolution is one word and can be up to byte address 0xFFFF (0x7FFF words). A value of zero indicates no RO area.

6.1.1.44 Start of RO Area Pointer (protected area) - 0x002D

Bits	Field Name	Default NVM Value	Description
15:11	Reserved	0x0	Reserved
10:0	SORO_area	0x0	Defines the start of the area in the EEPROM that is RO in word resolution See section 6.1.5 .

6.1.1.45 Watchdog Configuration - 0x002E

Bits	Field Name	Default NVM Value	Description
15	Watchdog Enable	0x0	Enable watchdog interrupt. See section 8.14.1 Valid values are: 0x0 Disabled 0x1 Enabled
14:11	Watchdog Timeout	0x2	Watchdog timeout period (in seconds). See section 8.14.1 Note: Loaded to 4 LSB bits of WDSTP.WD_Timeout field.
10:0	Reserved	0x0	Reserved



6.1.1.46 VPD Pointer - 0x002F

This word points to the Vital Product Data (VPD) structure. This structure is available for the NIC vendor to store it's own data. A value of 0xFFFF indicates that the structure is not available.

Bits	Field Name	Default NVM Value	Description
15:0	VPD offset	0x0	Offset to VPD structure in words. Bit 15 must be set to 0b (VPD area must be in the first 32 Kbytes of the EEPROM).For VPD Module inner structure See section 6.1.23

6.1.1.47 Setup Options PCI Function - 0x0030

The main setup options are stored in word 30h. These options are those that can be changed by the user via the Control-S setup menu.

Bits	Field Name	Default NVM Value	Description
15:13	RFU	0x0	Reserved. Must be 0.
12:10	FSD	0x0	Bits 12-10 control forcing speed and duplex during driver operation. Valid values are: 0x0 Auto-negotiate 0x1 10Mb/s Half Duplex 0x2 100Mb/s Half Duplex 0x3 3 Not valid (treated as 000b) 0x4 4 Not valid (treated as 000b) 0x5 5 10Mb/s Full Duplex 0x6 6 10Mb/s Full Duplex 0x7 1000Mb/s Full Duplex
9	RFU	0x0	
8	DSM	0x1	Display Setup Message. If the bit is set to 1, the "Press Control-S" message is displayed after the title message. Default value is 1.
7:6	PT	0x0	Prompt Time. These bits control how long the CTRL-S setup prompt message is displayed, if enabled by DIM Valid values are: 0x0 2 seconds (default) 0x1 3 seconds 0x2 5 seconds 0x3 0 seconds
5	DEP	0x0	Deprecated. Must be 0.
4:3	DBS	0x0	Default Boot Selection. These bits select which device is the default boot device. These bits are only used if the agent detects that the BIOS does not support boot order selection or if the MODE field of word 31h is set to MODE_LEGACY. Valid values are: 0x0 Network boot, then local boot 0x1 Local boot, then network boot 0x2 Network boot only 0x3 Local boot only
2:0	PS	0x0	Protocol Select. Valid values are: 0x0 PXE enabled 0x1 PXE disabled, possible iSCSI boot 0x2 Primary iSCSI Boot 0x3 Secondary iSCSI Boot Else Reserved

6.1.1.48 Configuration Customization Options PCI Function - 0x0031

Bits	Field Name	Default NVM Value	Description
15:14	SIG	0x1	Signature. Must be set to 01 to indicate that this word has been programmed by the agent or other configuration software.
13	RFU	0x0	Reserved. Must be 0.

Bits	Field Name	Default NVM Value	Description
12	RFU	0x0	Reserved. Must be 0.
11	RETRY	0x0	Selects Continuous Retry operation. If this bit is set, IBA will 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 will restart the PXE boot process again. If this bit is set, the only way to cancel PXE boot is for the user to press ESC on the keyboard. Retry will not be attempted due to hardware conditions such as an invalid EEPROM checksum or failing to establish link.
10:8	MODE	0x0	Selects the agent's boot order setup mode. This field changes the agent's 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: 0x0 Normal behavior. 0x1 Force Legacy mode. 0x2 Force BBS mode. 0x3 Force PnP Int18 mode. 0x4 Force PnP Int19 mode. 0x5 Reserved for future use.1 0x6 Reserved for future use.2 0x7 Reserved for future use.3
7	RFU	0x0	Reserved. Must be 0.
6	RFU	0x0	Reserved. Must be 0.
5	DFU	0x0	Disable Flash Update. If this bit is set to 1, the user is not allowed to update the flash image using PROSet. Default value is 0.
4	DLWS	0x0	Disable Legacy Wakeup Support. If this bit is set to 1, the user is not allowed to change the Legacy OS Wakeup Support menu option. Default value is 0.
3	DBS	0x0	Disable Boot Selection. If this bit is set to 1, the user is not allowed to change the boot order menu option. Default value is 0.
2	DPS	0x0	Disable Protocol Select. If set to 1, the user is not allowed to change the boot protocol. Default value is 0.
1	DTM	0x0	Disable Title Message. If this bit is set to 1, 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 wish the boot agent to display any messages at system boot. Default value is 0.
0	DSM	0x0	Disable Setup Menu. If this bit is set to 1, the user is not allowed to invoke the setup menu by pressing Control-S. In this case, the EEPROM may only be changed via an external program. Default value is 0.

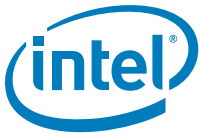
6.1.1.49 PXE Version - 0x0032

When the Boot Agent loads, it can check this value to determine if any first-time configuration needs to be performed. The 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	MAJ	0x1	PXE Boot Agent Major Version. Default value is 0.
11:8	MIN	0x3	PXE Boot Agent Minor Version. Default value is 0.
7:0	BLD	0x32	PXE Boot Agent Build Number. Default value is 0.

6.1.1.50 IBA Capabilities - 0x0033

This word 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 IBA.



Bits	Field Name	Default NVM Value	Description
15:14	SIG	0x1	Signature. Must be set to 01 to indicate that this word has been programmed by the agent or other configuration software.
13:5	RFU	0x0	Reserved. Must be 0.
4	ISCSI	0x0	iSCSI Boot is present in flash if set to 1.
3	UEFI	0x0	UEFI UNDI driver is present in flash if set to 1.
2	RPL	0x0	RPL module is present in flash if set to 1.
1	UNDI	0x1	PXE UNDI driver is present in flash if set to 1.
0	BC	0x1	PXE Base Code is present in flash if set to 1.

6.1.1.51 PCIe PHY Configuration 0 Low - 0x0034

Bits	Field Name	Default NVM Value	Description
15:9	TX_SWING_LOW	0x3F	Transmit amplitude
8	VREG_BYPASS	0x0	
7	REF_CLK_PAD	0x1	
6	REF_CLK_DIV2	0x0	
5:0	TX_DE_EMPH_6DB	0x23	Gen 2 6db de-emphasis

6.1.1.52 PCIe PHY Configuration 0 High - 0x0035

Bits	Field Name	Default NVM Value	Description
15:13	RXN_REQ	0x2	
12:8	TX_TERM_OFFSET	0x0	
7	CMCLK_SEL	0x0	
6	Reserved	0x0	
5:0	TX_DE_EMPH_3DB	0x1C	Gen 2 3.5db de-emphasis

6.1.1.53 iSCSI Option ROM Version - 0x0036

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Option ROM Version	0xFFFF	

6.1.1.54 Alternate MAC Address Location - 0x0037

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address Location	0x0	

Field Alternate MAC Address Location points to Alternate MAC Address Section. For Alternate MAC Address Section inner structure See section 6.1.12

6.1.1.55 PCIe PHY Configuration 1 Low - 0x0038

Bits	Field Name	Default NVM Value	Description
15:14	Reserved	0x0	
13	PORT_SEL	0x1	
12	Reserved	0x0	
11:9	TX_VBOOST	0x5	Transmit launch amplitude
8	G2_DIS_OVRD	0x0	0b - PCIe Gen2 is enabled and the TS1 ordered Set Symbol 4 equals to 0x6. 1b - PCIe Gen2 is disabled and the TS1 ordered Set Symbol 4 equals to 0x2.
7	TX_NO_DE_EMPH	0x0	
6:0	MPLL_M	0x19	

6.1.1.56 PCIe PHY Configuration 1 High - 0x0039

Bits	Field Name	Default NVM Value	Description
15:9	TX_SWING_FULL	0x6F	
8:6	Reserved	0x0	
5:0	TX_DE_EMPH_GEN1	0x1C	

6.1.1.57 PCIe PHY Configuration 2 Low / Reset to PCIe PHY Delay (x40us) - 0x003A

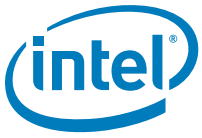
Bits	Field Name	Default NVM Value	Description
15:8	PCI_PHY_RST_DELAY	0x0	Loaded to the PCIE_RST_DELAY field in the PCIE_T2PRST register
7:3	LOS_LVL	0x6	Loaded to the LOS_LVL field in the PCIe_PHY_CFG2 register
2:0	LOS_BIAS	0x4	Loaded to the LOS_BIAS field in the PCIe_PHY_CFG2 register

6.1.1.58 Reserved - 0x003B

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	

6.1.1.59 PXE VLAN pointer - 0x003C

Bits	Field Name	Default NVM Value	Description
15:0	PXE VLAN pointer	0x0	Pointer to PXE VLAN configuration section. Multiple of 4K bytes (2K words). See section 6.1.4 .



6.1.1.60 iSCSI Boot Configuration Pointer - 0x003D

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Address	0x0	Pinter to the iSCSI Bot Configuration section. If set to 0x0000h or 0xFFFF there is no EEPROM configuration data available for the iSCSI adapter. In this case configuration data must be provided by the BIOS through the SM CLP interface. See section 6.1.14

6.1.1.61 Reserved - 0x003E

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFFFF	

6.1.1.62 Checksum Word - 0x003F

Bits	Field Name	Default NVM Value	Description
15:0	Checksum Word	0x0	The checksum words (Offset 0x3F from start of Common, LAN 1, LAN 2 and LAN 3 sections) are used to ensure that the base EEPROM image is a valid image. The value of this word should be calculated such that after adding all the words (0x00:0x3E), including the checksum word itself, the sum should be 0xBABA. The initial value in the 16-bit summing register should be 0x0000 and the carry bit should be ignored after each addition. Note: Hardware does not calculate the checksum word during EEPROM write; it must be calculated by software independently and included in the EEPROM write data. Hardware does not compute a checksum over words 0x00:0x3F during EEPROM reads in order to determine validity of the EEPROM image; this field is provided strictly for software verification of EEPROM validity. All hardware configurations based on word 0x00:0x3F content is based on the validity of the Signature field of the EEPROM Sizing & Protected Fields EEPROM word (Signature must be 01b).

6.1.1.63 Free Provisioning Area Pointer - 0x0040

Bits	Field Name	Default NVM Value	Description
15	4K Ptr type mark	0x1	
14:0	Free Provisioning Area Pointer	0x0	

Field Free Provisioning Area Pointer points to Free Provisioning Area Section. For Free Provisioning Area inner structure See section 6.1.26

6.1.1.64 Free Provisioning Area Size - 0x0041

Bits	Field Name	Default NVM Value	Description
15:0	Free Provisioning Area Size in 4KB	0xFB	The Free Provisioning Area Size for an NVM image that does not includes an OROM image is 0x7B (492 KB). And the Free Provisioning Area Size for an NVM image that includes an OROM image is 0xFB (1004 KB).

6.1.1.65 Image Unique ID 0 - 0x0042

These words contain a unique 32-bit ID for each image generated by Intel to enable tracking of images and comparison to the original image if testing a customer EEPROM image.

Bits	Field Name	Default NVM Value	Description
15:0	Image Unique ID		

6.1.1.66 Image Unique ID 1 - 0x0043

These words contain a unique 32-bit ID for each image generated by Intel to enable tracking of images and comparison to the original image if testing a customer EEPROM image.

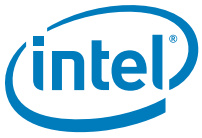
Bits	Field Name	Default NVM Value	Description
15:0	Image Unique ID		

6.1.1.67 PCIe L1 Substates Capability Low - 0x0044

Bits	Field Name	Default NVM Value	Description
15:8	Tcommon-Mode	0x37	Time (in us) required for this Port to re-establish common mode (0-255 us), when exiting L1.2.
7:5	Reserved	0x0	
4	L1 PM Substates Supported	0x1	This bit is loaded to the "L1 PM Substates Supported" bit field in the "L1 PM Sub states Capabilities Register" register in the PCIe configuration space.
3	ASPM L1.1 Supported	0x1	This bit is loaded to the "ASPM L1.1 Supported" bit field in the "L1 PM Sub states Capabilities Register" register in the PCIe configuration space.
2	ASPM L1.2 Supported	0x1	This bit is loaded to the "ASPM L1.2 Supported" bit field in the "L1 PM Sub states Capabilities Register" register in the PCIe configuration space.
1	PCI-PM L1.1 Supported	0x1	This bit is loaded to the "PCI-PM L1.1 Supported" bit field in the "L1 PM Sub states Capabilities Register" register in the PCIe configuration space.
0	PCI-PM L1.2 Supported	0x1	This bit is loaded to the "PCI-PM L1.2 Supported" bit field in the "L1 PM Sub states Capabilities Register" register in the PCIe configuration space.

6.1.1.68 PCIe L1 Substates Capability High - 0x0045

Bits	Field Name	Default NVM Value	Description
15	L1 Substate Enable	1	Expose the L1 substates capability structure in the PCIe configuration space
14:8	Reserved	0x0	
7:3	Port T_POWER_ON Value	0x9	The time (in us) that is requires from the PCIe link partner to wait in L1.2. Exit after sampling CLKREQ# before actively driving the interface. The wait time equals to "Port T_POWER_ON" multiplied by the "Port T_POWER_ON Scale".
2	Reserved	0x0	



Bits	Field Name	Default NVM Value	Description
1:0	Port T_POWER_ON Scale	0x0	Specifies the scale for the "Port T_POWER_ON Value" as follow: 00b = 2 us 01b = 10 us 10b = 100 us 11b = Reserved

6.1.1.69 PCIe L1 Substates Control 1st Low - 0x0046

Bits	Field Name	Default NVM Value	Description
15:8	Common Mode Restore Time	0x0	
7:4	Reserved	0x0	
3	ASPM L1.1 Enable	0x0	
2	ASPM L1.2 Enable	0x0	
1	PCI-PM L1.1 Enable	0x0	
0	PCI-PM L1.2 Enable	0x0	

6.1.1.70 PCIe L1 Substates Control 1st High - 0x0047

Bits	Field Name	Default NVM Value	Description
15:13	LTR_L1.2_THRESHOLD_Scale	0x0	Provides a scale for the "LTR_L1.2_THRESHOLD_Value". Default setting is 0x0 (nsec units)
12:10	Reserved	0x0	
9:0	LTR_L1.2_THRESHOLD_Value	0x0	Indicates the LTR threshold used to determine if entry into L1 results in L1.1 (if enabled) or L1.2 (if enabled).

6.1.1.71 PCIe L1 Substates Control 2nd (only lower byte taken) - 0x0048

Bits	Field Name	Default NVM Value	Description
15:11	CLK warm up Value	0x1	Loaded to the PCIE_L1_EXTCLK -> LTSSM_L1EXTCLK_WARMUP_Value.
10	Reserved	0x0	
9:8	CLK warmup_scale	0x1	Loaded to the PCIE_L1_EXTCLK -> LTSSM_L1EXTCLK_WARMUP_Scale: 00b = 2µs, 01b = 10µs, 10b = 100µs, 11b = Reserved.
7:3	T_POWER_ON Value	0x9	The minimum amount of time (in us) that the Port must wait in L1.2.Exit after sampling CLKREQ# before actively driving the interface.
2	Reserved	0x0	
1:0	T_POWER_ON Scale	0x0	Specifies the scale used for T_POWER_ON Value. Default setting is 0x0 00b = 2us 01b = 10us 10b = 100us11b = Reserved

6.1.1.72 PTM Setting - 0x0049

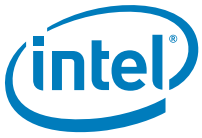
Bits	Field Name	Default NVM Value	Description
15:8	PTM Effective Granularity	0x4	Loaded to the Local Clock Granularity field in the PTM Capability Register in the PCIe configuration space
7:4	Reserved	0x0	
3:2	PTM Endian Notation	0x0	Endian Notation of the "Propagation Delay" in the PTM ResponseD message: 00b - Little Endian 01b - Big Endian 10b - Unknown, Foxville auto-detects the Endianess notation 11b - Reserved
1	PTM Requester Capable	0x1	Loaded to the "PTM Requester Capable" flag in the "PTM Capability Register" in the PCIe configuration space.
0	PTM Enable	0x0	This bit field is loaded to the "PTM capability Enabled" flag in the PCIe configuration space. It should be set to '0' from the NVM and the OS is expected to set it to '1'.

6.1.1.73 EXP. ROM Boot Code Section Pointer - 0x004A

Bits	Field Name	Default NVM Value	Description
15	4K Ptr type mark	0x1	
14:12	Reserved	0x0	
11:0	EXP. ROM Pointer	0x0	Pointer to start address of EXP. ROM section. Loaded to the 12 MS bits of the PCIE_EXP_ROM_BASE field in the PCIE_EXP_ROM_BASE register. See section 6.1.18

6.1.1.74 ULP Capability Enable - 0x004B

Bits	Field Name	Default NVM Value	Description
15	ULP_EN_DevOff	0x1	Enable Ultra Low Power state mode of operation by the LAN_DISABLE_N signal.
14	Reserved	0x0	
13	ENGY_Wake_DevOff	0x0	Wake on link energy in ULP state triggered by the LAN_DISABLE_N signal. 0b - Wake on Link Energy is inactive 1b - Wake on Link Energy is active
12	BCON_EN_DevOff		Enable Beacons transmission while in the ULP state following DEVOFF signal
11	BCON_EN_Dr		Enable Beacons transmission while in the ULP state initiated by the OS
10	LinkOff_EN_InAct	0x1	Enable Link Disconnect in Dx state if the link is not needed or when the device is disabled.
9:8	Reserved	0x7	Reserved
7:4	Reserved	0xF	Reserved
3:2	Reserved	0x3	Reserved
1:0	ULP_Delay	0x2	Delay from PE_RST_N driven low till ULP is activated. The delay is defined in seconds units.



6.1.1.75 Reserved - 0x004C

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xFF	Reserved

6.1.1.76 Reserved - 0x004D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.77 RO Commands Version - 0x0050

Bits	Field Name	Default NVM Value	Description
15:0	RO Commands Version	0x0	

6.1.1.78 Firmware Module Configuration Pointer - 0x0051

MNG

Bits	Field Name	Default NVM Value	Description
15:0	Firmware Patch Pointer	0x0	Pointer to loader patch structure. See section 6.1.2

6.1.1.79 PCIe PHY Configuration 3 Low - 0x0052

Bits	Field Name	Default NVM Value	Description
15:0	L12_T_PO	0x1F	This field multiplied by 8 is loaded to the LTSSM_L12_T_PO field in the PCIE_L12_TIMES register bits 3:10 of respectively.
7:0	L11_T_PO	0x05	Loaded to the LTSSM_L11_T_PO field in the PCIE_L11_TIMES register bits 3:10 of respectively.

6.1.1.80 PCIe PHY Configuration 3 Hi - 0x0053

Bits	Field Name	Default NVM Value	Description
15:8	L12_STAY	0x0F	Loaded to the LTSSM_L12_STAY field in the PCIE_L12_TIMES register bits 19:26 of respectively.

Bits	Field Name	Default NVM Value	Description
7:0	RSVD	0xFF	Reserved

6.1.1.81 PCIe PHY Configuration 4 Low - 0x0054

Bits	Field Name	Default NVM Value	Description
15:8	P11_2_P10	0x3F	This field multiplied by 4 is loaded to the PHY_P11_2_P10 field in the PCIE_TIMES_2P10 register bits 2:9 of respectively.
7:0	P12_2_P10	0x3F	This field multiplied by 4 is loaded to the PHY_P12_2_P10 field in the PCIE_TIMES_2P10 register bits 18:25 of respectively.

6.1.1.82 PCIe PHY Configuration 4 High - 0x0055

Bits	Field Name	Default NVM Value	Description
15:0	High	0xffff	16 MS bits of the PCIe PHY Configuration 4 register

6.1.1.83 PCIe PHY Configuration 5 Low - 0x0056

Bits	Field Name	Default NVM Value	Description
15:0	Low	0xffff	16 LS bits of the PCIe PHY Configuration 5 register

6.1.1.84 PCIe PHY Configuration 5 High - 0x0057

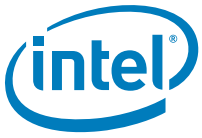
Bits	Field Name	Default NVM Value	Description
15:0	High	0xffff	16 MS bits of the PCIe PHY Configuration 5 register

6.1.1.85 Reserved - 0x0058

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.86 Reserved - 0x0059

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	



6.1.1.87 Reserved - 0x005A

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.88 Reserved - 0x005B

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.89 Reserved - 0x005C

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.90 Reserved - 0x005D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.91 Reserved - 0x005E

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.92 Reserved - 0x005F

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.93 Reserved - 0x0060

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.94 Reserved - 0x0061

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.95 Reserved - 0x0062

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.96 Reserved - 0x0063

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.97 Reserved - 0x0064

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.98 Reserved - 0x0065

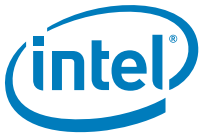
Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.99 Reserved - 0x0066

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.100 Reserved - 0x0067

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	



6.1.1.101 Reserved - 0x0068

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.102 Reserved - 0x0069

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.103 Reserved - 0x006A

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.104 Reserved - 0x006B

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.105 Reserved - 0x006C

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.106 Reserved - 0x006D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.107 Reserved - 0x006E

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.108 Reserved - 0x006F

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.109 Reserved - 0x0070

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.110 Reserved - 0x0071

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.111 Reserved - 0x0072

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.112 Reserved - 0x0073

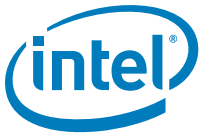
Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.113 Reserved - 0x0074

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.114 Reserved - 0x0075

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	



6.1.1.115 Reserved - 0x0076

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.116 Reserved - 0x0077

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.117 Reserved - 0x0078

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.118 Reserved - 0x0079

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.119 Reserved - 0x007A

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.120 Reserved - 0x007B

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.121 Reserved - 0x007C

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.122 Reserved - 0x007D

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.123 Reserved - 0x007E

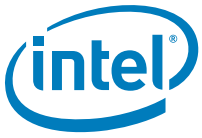
Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.1.124 Reserved - 0x007F

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0xffff	

6.1.2 Firmware Module Configuration Section Summary Table

Word Offset	Description	Reference
0x0000	Reserved 0x0	See section 6.1.2.1
0x0001	Reserved 0x1	See section 6.1.2.2
0x0002	Reserved 0x2	See section 6.1.2.3
0x0003	Common Firmware Parameters pointer	See section 6.1.2.4
0x0004	Reserved 0x4	See section 6.1.2.5
0x0006	SideBand Configuration Pointer	See section 6.1.2.6
0x0007	Flexible TCO Filter Configuration Pointer	See section 6.1.2.7
0x0008	Reserved 0x8	See section 6.1.2.8
0x0009	Reserved 0x9	See section 6.1.2.9
0x000A	Reserved - NC-SI Configuration Pointer	See section 6.1.2.10
0x000B	Traffic Types Parameters Pointer	See section 6.1.2.11
0x000C	Reserved	
0x000D	Reserved 0xD	See section 6.1.2.12
0x000E	Reserved 0xE	See section 6.1.2.13
0x000F	PHY Configuration Pointer	See section 6.1.2.14
0x0010	Reserved	



6.1.2.1 Reserved 0x0 - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Reserved 0x0		

6.1.2.2 Reserved 0x1 - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Reserved 0x1	0xFFFF	

6.1.2.3 Reserved 0x2 - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Reserved 0x2	0xFFFF	

6.1.2.4 Common Firmware Parameters pointer - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	ptr	0x0	Field ptr points to Common Firmware Parameters Section. For Common Firmware Parameters inner structure See section 6.1.3

6.1.2.5 Reserved 0x4 - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	Reserved 0x4		

6.1.2.6 SideBand Configuration Pointer - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	SideBand Configuration Pointer	0x0	This field points to SideBand Configuration Modules Section. See section 6.1.8 for a description of this section.

6.1.2.7 Flexible TCO Filter Configuration Pointer - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	Flexible TCO Filter Configuration Pointer	0x0	This field points to TCO Filter Configuration Modules Section. See section 6.1.9 for a description of this section.

6.1.2.8 Reserved 0x8 - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	Reserved 0x8		

6.1.2.9 Reserved 0x9 - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	Reserved 0x9		

6.1.2.10 Reserved - NC-SI Configuration Pointer - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	Reserved - NC-SI Configuration Pointer	0x0	

6.1.2.11 Traffic Types Parameters Pointer - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	Traffic Types Parameters	0x0	This Field points to Traffic Types Parameters Section. For Traffic Types Parameters inner structure See section 6.1.10

6.1.2.12 Reserved 0xD - 0x000D

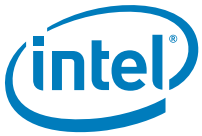
Bits	Field Name	Default NVM Value	Description
15:0	Reserved 0xD		

6.1.2.13 Reserved 0xE - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	Reserved 0xE	0xFFFF	

6.1.2.14 PHY Configuration Pointer - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	PHY Configuration Pointer	0x0	This Field points to PHY Configuration Section. For PHY Configuration inner structure See section 6.1.11



6.1.3 Common Firmware Parameters Section Summary Table

Word Offset	Description	Reference
0x0000	Section header	See section 6.1.3.1
0x0001	Common Firmware Parameters 1	See section 6.1.3.2
0x0002	Common Firmware Parameters 2	See section 6.1.3.3

6.1.3.1 Section header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:8	Block CRC		CRC-8-CCITT: Start Section -> Word: Common Firmware Parameters -> Section header End Section -> Word: Common Firmware Parameters -> Common Firmware Parameters 2
7:0	Block Length		Length in words of the section covered by CRC

6.1.3.2 Common Firmware Parameters 1 - 0x0001

Bits	Field Name	Default NVM Value	Description
15	Enable Firmware Reset	0x1	0b = Firmware reset via HICR is disabled. 1b = Firmware reset via HICR is enabled.NVM Valid values are: 0x0 Disabled 0x1 Enabled
14:13	Sideband Interface	0x3	00b = Reserved. 01b = Reserved. 10b = Pass Through traffic only over SMBus. 11b = Pass Through traffic over SMBus or PCIe. Relevant only if Manageability Mode is set to Enabled.
12	Restore MAC Address	0x0	0x0 Do not restore MAC address at power on 0x1 Restore MAC address at power on
11	Reserved	0x1	
10:8	Manageability Mode	0x0	Valid values are: 0x0 None 0x1 Reserved 0x2 Enabled 0x3 Reserved 3 0x4 Reserved 4 0x5 Reserved 5 0x6 Reserved 6 0x7 Reserved 7
7:5	Reserved	0x7	
4	LAN Force TCO Reset Disable	0x1	0x0 Enable Force TCO reset on LAN 0x1 Disable Force TCO reset on LAN
3	Proxying Capable	0x1	0x0 Disable Protocol Offload 0x1 Enable Protocol Offload
2	OS2BMC Capable	0x0	0x0 Disabled 0x1 Enabled

Bits	Field Name	Default NVM Value	Description
1	Allow SW Programming of RAM Base Address	0x0	0x0 Disabled 0x1 Enabled
0	res	0x1	

6.1.3.3 Common Firmware Parameters 2 - 0x0002

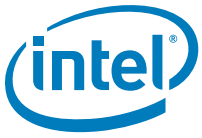
Bits	Field Name	Default NVM Value	Description
15:12	Reserved		Reserved
11	Reserved	0x0	Multi-Drop NC-SI Topology. 0b = Point-to-point. 1b = Multi-drop (default). When this bit is set, the NCSI_CRS_DV and NCSI_RXD[1:0] pins are High-Z following power-up; otherwise, the pins are driven.
10	PARITY_ERR_RST_EN	0x1	When set, enables reset of management logic and generation of internal firmware reset as a result of parity error detected in management memories. 0x0 Disabled 0x1 Enabled
9	Reserved	0x1	Reserved
8:6	Reserved		Reserved
5:1	Semaphore Backoff Interval	0x1	Number of 10 ms ticks that firmware must wait before taking semaphore ownership again since it has released it.
0	LAN_FTCO_ISOL_DIS	0x1	LAN Force TCO Isolate Disable 1b = Disable. 0b = Enable.

6.1.4 PXE VLAN Configuration Section Summary Table

Word Offset	Description	Reference
0x0000	VLAN Block Signature	See section 6.1.4.1
0x0001	Version and Size	See section 6.1.4.2
0x0002	Port 0 VLAN Tag	See section 6.1.4.3

6.1.4.1 VLAN Block Signature - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	VLAN Block Signature	0x4C56	ASCII 'V', 'L'



6.1.4.2 Version and Size - 0x0001

Bits	Field Name	Default NVM Value	Description
15:8	Size		Length in byte units First Section -> Word: PXE VLAN Configuration Section -> VLAN Block Signature Last Section -> Word: PXE VLAN Configuration Section -> Port 0 VLAN Tag total size in bytes of section
7:0	Version	0x01	

6.1.4.3 Port 0 VLAN Tag - 0x0002

Bits	Field Name	Default NVM Value	Description
15:13	Priority (0-7)	0x0	
12	reserved	0x0	
11:0	VLAN ID (1- 4095)	0x0	

6.1.5 RO Start Section Summary Table

Word Offset	Description	Reference
0x0000 (see text)	Dummy word. The address of this block is defined by the "Start of RO Area pointer" word at address 0x002D	See section 6.1.5.1

6.1.5.1 Dummy word - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Dummy word	0xffff	

6.1.6 LPG Reset CSR Auto Configuration Section Summary Table

This word points to the CSR auto configuration power-up structure of the LAN that is read only following power up. Sections are loaded during HW auto-load. If no CSR auto-load is required, the word must be set to 0xFFFF

Word Offset	Description	Reference
0x0000	Section Length	See section 6.1.6.1
0x0001	Block CRC8	See section 6.1.6.2
0x0002	PCIEERRCTL address 5BA0	See section 6.1.6.3
0x0003	PCIEERRCTL LSB	See section 6.1.6.4
0x0004	PCIEERRCTL Data MSB	See section 6.1.6.5
0x0005	PCIEPHYDAT address 4 0x5B44	See section 6.1.6.6

Word Offset	Description	Reference
0x0006	PCIEPHYDAT data 0xA800A8	See section 6.1.6.7
0x0007	PCIEPHYDAT data 4 MS	See section 6.1.6.8
0x0008	PCIEPHYADR address 5 0x5B40	See section 6.1.6.9
0x0009	PCIEPHYADR data 0x5E000090	See section 6.1.6.10
0x000A	PCIEPHYADR data 5 MS	See section 6.1.6.11
0x000B	Device Control Register address 0	See section 6.1.6.12
0x000C	Device Control Register Data LSB	See section 6.1.6.13
0x000D	Device Control Register Data MSB	See section 6.1.6.14
0x000E	EEE_SU address	See section 6.1.6.15
0x000F	EEE_SU Data LSB	See section 6.1.6.16
0x0010	EEE_SU Data MSB	See section 6.1.6.17
0x0011	CSR Address	See section 6.1.6.18
0x0012	CSR Data LSB	See section 6.1.6.19
0x0013	CSR Data MSB	See section 6.1.6.20

6.1.6.1 Section Length - 0x0000

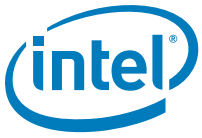
Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	
14:0	Section Length		Length in words of the section covered by CRC (excluding the length word and the CRC word). Section length = (3 * number of CSRs to be configured).

6.1.6.2 Block CRC8 - 0x0001

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: LPG Reset CSR Auto Configuration -> PCIEERRCTL address 5BA0 End Section -> Word: LPG Reset CSR Auto Configuration -> CSR Data MSB CRC8 is computed over the module, header excluded (for example, starting from word offset 0x2 included).

6.1.6.3 PCIEERRCTL Address 5BA0 - 0x0002

Bits	Field Name	Default NVM Value	Description
15	Reserevd	0x0	
14:0	PCIEERRCTL add	0x16e8	



6.1.6.4 PCIEERRCTL LSB - 0x0003

Bits	Field Name	Default NVM Value	Description
15:14	reserved	0x0	
13	ERR INJ RX CDQ 3	0x0	
12	ERR EN RX CDQ 3	0x1	
11	ERR INJ RX CDQ 2	0x0	
10	ERR EN RX CDQ 2	0x1	
9	ERR INJ RX CDQ 1	0x0	
8	ERR EN RX CDQ 1	0x1	
7	ERR INJ RX CDQ 0	0x0	
6	ERR EN RX CDQ 0	0x1	
5:1	Reserved	0x0	
0	GPAR_EN	0x1	

6.1.6.5 PCIEERRCTL Data MSB - 0x0004

Bits	Field Name	Default NVM Value	Description
15	PCIe Flush	0x0	
14:0	reserved	0x0	

6.1.6.6 PCIEPHYDAT Address 4 0x5B44 - 0x0005

Bits	Field Name	Default NVM Value	Description
15	Reserevd	0x0	
14:0	PCIEERRCTL add	0x16d1	

6.1.6.7 PCIEPHYDAT Data 0xA800A8 - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	data	0xA800A8	

6.1.6.8 PCIEPHYDAT Data 4 MS - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	MS_word		

6.1.6.9 PCIEPHYADR Address 5 0x5B40 - 0x0008

Bits	Field Name	Default NVM Value	Description
15	Reserevd	0x0	
14:0	PCIEERRCTL add	0x16d0	

6.1.6.10 PCIEPHYADR Data 0x5E000090 - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	data	0x5E000090	wr_en=1; byte_en=1111b

6.1.6.11 PCIEPHYADR Data 5 MS - 0x000A

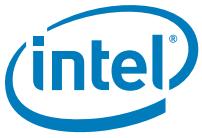
Bits	Field Name	Default NVM Value	Description
15:0	MS_word		

6.1.6.12 Device Control Register Address 0 - 0x000B

Bits	Field Name	Default NVM Value	Description
15	Reserevd	0x0	
14:0	PCIEERRCTL add	0x0	

6.1.6.13 Device Control Register Data LSB - 0x000C

Bits	Field Name	Default NVM Value	Description
15:13	reserved	0x0	
12	FRCDPLX	0x0	Force Duplex. When set to 1b, software can override the duplex indication from the PHY that is indicated in the FDX to the MAC. Otherwise, the duplex setting is sampled from the PHY FDX indication into the MAC on the asserting edge of the PHY LINK signal. When asserted, the CTRL.FD bit sets duplex.
11	FRCSPD	0x0	Force Speed. This bit is set when software needs to manually configure the MAC speed settings according to the SPEED bits.
10:8	SPEED	0x0002	Speed Selection. These bits determine the speed configuration and are written by software after reading the PHY configuration through the MDIO interface. These signals are ignored when auto-speed detection is enabled. 000b = 10 Mb/s. 001b = 100 Mb/s. 010b = 1000 Mb/s. 110b = .
7	Reserved	0x0	Reserved 0x0
6	SLU	0x1	Set Link Up. It must be set to 1b permitting the MAC to recognize the LINK signal from the PHY, which indicates the PHY has gotten the link up, and is ready to receive and transmit data.
5:3	Reserved	0x0	



Bits	Field Name	Default NVM Value	Description
2	GIO Master Disable	0x0	When set to 1b, the function of this bit blocks new master requests including manageability requests. If no master requests are pending by this function, the STATUS.GIO Master Enable Status bit is set. See section 5.2.3.3 for further information.
1	Reserved	0x0	Reserved
0	FD	0x1	Full-Duplex. Controls the MAC duplex setting when explicitly set by software. 0b = Half duplex. 1b = Full duplex.

6.1.6.14 Device Control Register Data MSB - 0x000D

Bits	Field Name	Default NVM Value	Description
15	PHY_RST	0x0	PHY Reset. Generates a hardware-level reset to the internal PHY. 0b = Normal operation. 1b = Internal PHY reset asserted.
14	VME	0x0	VLAN Mode Enable. When set to 1b, VLAN information is stripped from all received 802.1Q packets.
13	DEV_RST (SC)	0x0	Device Reset. This bit performs a reset of the entire controller device, resulting in a state nearly approximating the state following a power-up reset or internal PCIe reset, except for system PCI configuration. 0b = Normal. 1b = Reset.
12	TFCE	0x0	Transmit Flow Control Enable. When set, indicates that Foxville transmits flow control packets (XON and XOFF frames) based on the receiver fullness. If auto-negotiation is enabled, this bit is set to the negotiated duplex value.
11	RFCE	0x1	Receive Flow Control Enable. When set, indicates that Foxville responds to the reception of flow control packets. If auto-negotiation is enabled, this bit is set to the negotiated flow control value.
10	Reserved	0x0	Reserved
9:8	Reserved	0x0	
7	SDP1_IODIR	0x0	SDP1 Pin Direction. Controls whether software-controllable pin SDP1 is configured as an input or output (0b = input, 1b = output).
6	SDP0_IODIR	0x0	SDP0 Pin Direction. Controls whether software-controllable pin SDP0 is configured as an input or output (0b = input, 1b = output).
5	SDP0_WDE	0x0	SDP0 used for Watchdog Indication. When set, SDP0 is used as a watchdog indication. When set, the SDP0_DATA bit indicates the polarity of the watchdog indication. In this mode, SDP0_IODIR must be set to an output.
4	ADVD3WUC	0x1	D3Cold Wake up Capability Enable. When this bit is set to 0b, PME (WAKE#) is not generated in D3Cold.
3	SDP1 DATA	0x0	SDP1 Data Value. Used to read or write the value of software-controlled I/O pin SDP1. If SDP1 is configured as an output (SDP1_IODIR = 1b), this bit controls the value driven on the pin (initial value NVM-configurable). If SDP1 is configured as an input, reads return the current value of the pin.
2	SDP0 DATA	0x0	SDP0 Data Value. Used to read or write the value of software-controlled I/O pin SDP0. If SDP0 is configured as an output (SDP0_IODIR = 1b), this bit controls the value driven on the pin (initial value NVM-configurable). If SDP0 is configured as an input, reads return the current value of the pin. When the SDP0_WDE bit is set, this field indicates the polarity of the watchdog indication.
1	SDP1_GPIEN	0x0	General Purpose Interrupt Detection Enable for SDP1. If software-controlled I/O pin SDP1 is configured as an input, this bit (when 1b) enables the use for GPI interrupt detection.
0	SDP0_GPIEN	0x0	General Purpose Interrupt Detection Enable for SDP0. If software-controlled I/O pin SDP0 is configured as an input, this bit (when 1b) enables the use for GPI interrupt detection.

6.1.6.15 EEE_SU Address - 0x000E

Bits	Field Name	Default NVM Value	Description
15	Reserevd	0x0	
14:0	EEE_SU add	0x038d	

6.1.6.16 EEE_SU Data LSB - 0x000F

Bits	Field Name	Default NVM Value	Description
15:8	Tw_min_100	0x3c	
7:0	Tw_min_1000	0x21	

6.1.6.17 EEE_SU Data MSB - 0x0010

Bits	Field Name	Default NVM Value	Description
15:10	TEEE_DLY	0x2	
9:8	TX_LU_LPI_DLY	0x3	
7	LPI_Clock_Stop	0x0	
6	XXMII_LPI_Stop	0x1	
5:0	Tw_wake_min	0x0	

6.1.6.18 CSR Address - 0x0011

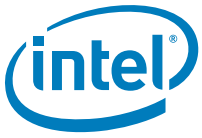
Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	
14:0	CSR_ADDR	0x0000	CSR Address in Double Words (4 bytes).

6.1.6.19 CSR Data LSB - 0x0012

Bits	Field Name	Default NVM Value	Description
15:0	CSR_Data_LSB	0x0000	CSR Data LSB.

6.1.6.20 CSR Data MSB - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	CSR_Data_MSB		



6.1.7 RO End Section Summary Table

Word Offset	Description	Reference
0x0000 (see text)	Dummy word. The address of this block is defined by the "End of RO Area pointer" word at address 0x002C	See section 6.1.7.1

6.1.7.1 Dummy word - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Dummy word	0xffff	

6.1.8 SideBand Configuration Section Summary Table

Word Offset	Description	Reference
0x0000	Section header	See section 6.1.8.1
0x0001	SMBus Maximum Fragment Size	See section 6.1.8.2
0x0002	SMBus Notification Timeout and Flags	See section 6.1.8.3
0x0003	SMBus Slave Addresses	See section 6.1.8.4
0x0004	Reserved 0x4	See section 6.1.8.5
0x0005	Reserved 0x5	See section 6.1.8.6
0x0006	NC-SI Configuration #1	See section 6.1.8.7
0x0007	NC-SI Configuration #2	See section 6.1.8.8
0x0008	Reserved	See section 6.1.8.9
0x0009	MCTP UUID - Time Low LSB	See section 6.1.8.10
0x000A	MCTP UUID - Time Low MSB	See section 6.1.8.11
0x000B	MCTP UUID - Time MID	See section 6.1.8.12
0x000C	MCTP UUID - Time High and Version	See section 6.1.8.13
0x000D	MCTP UUID - Clock Seq	See section 6.1.8.14
0x000E	MCTP UUID - Node1	See section 6.1.8.15
0x000F	MCTP UUID - Node2	See section 6.1.8.16
0x0010	MCTP UUID - Node3	See section 6.1.8.17
0x0011	Alternative IANA	See section 6.1.8.18
0x0012	NC_SI over MCTP Message Types	See section 6.1.8.19
0x0013	NC_SI over MCTP Configuration	See section 6.1.8.20
0x0014	MCTP rate limiter config 1	See section 6.1.8.21
0x0015	MCTP rate limiter config 2	See section 6.1.8.22

6.1.8.1 Section header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:8	Block CRC		CRC-8-CCITT: Start Section -> Word: SideBand Configuration -> Section header End Section -> Word: SideBand Configuration -> MCTP rate limiter config 2
7:0	Block Length		Length in words of the section covered by CRC

6.1.8.2 SMBus Maximum Fragment Size - 0x0001

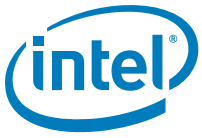
Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x00	
7:0	SMBus Maximum Fragment Size	0x45	

6.1.8.3 SMBus Notification Timeout and Flags - 0x0002

Bits	Field Name	Default NVM Value	Description
15:8	SMBus Notification	0xff	
7:6	SMBus Connection Speed	0x2	Valid values are: 0x0 100K Standard SMBus connection (100 KHz) 0x1 400K 400 KHz connection 0x2 1M 1 MHz connection 0x3 Reserved
5	SMBus Block Read	0x0	0x0 Block read command is 0xC0 0x1 Block read command is 0xD0
4	Enable Auto SMBus	0x0	When set, the device transits autonomously from PCIe VDM to SMBus when the BME is turned off or when the device transits to D3.
3	Enable fairness	0x1	0x0 Disabled 0x1 Enabled
2	Disable SMBus ARP	0x0	0x0 Enabled 0x1 Disabled
1	SMBus ARP PEC	0x1	0x0 Disabled 0x1 Enabled
0	SMBus Transaction PEC	0x0	0x0 Disabled 0x1 Enabled

6.1.8.4 SMBus Slave Addresses - 0x0003

Bits	Field Name	Default NVM Value	Description
15:9	MC SMBus slave address	0x62	The MC Slave address should be set to 0x62 (address 0xC4).
8	Reserved	0x0	
7:1	SMBus Slave Address	0x34 / 0x49	Foxville SMBus slave address. It should be set to 0x34 (address 0x68) for Docking station and 0x49 (address 0x92) for Non-Docking setup.
0	Reserved	0x0	



6.1.8.5 Reserved 0x4 - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	

6.1.8.6 Reserved 0x5 - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Fail-Over Register high	0x0	

6.1.8.7 NC-SI Configuration #1 - 0x0006

Bits	Field Name	Default NVM Value	Description
15:12	Reserved	0x0	
11	Reserved	0x0	
10	Reserved	0x0	
9	Reserved	0x0	
8	Reserved	0x0	
7:5	Package ID	0x0	
4:0	Reserved	0x0	

6.1.8.8 NC-SI Configuration #2 - 0x0007

Bits	Field Name	Default NVM Value	Description
15	Read Package ID From SDP	0x0	0b = Read from NVM. 1b = Read from SDP
14:0	Reserved	0x0	

6.1.8.9 Reserved - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0	NC-SI HW-Arbitration TOKEN Timeout (in 16 ns cycles). In order to get the value if NC-SI REF_CLK cycles, this field should be multiplied by 4/5. Setting value to 0 disables the timeout mechanism.

6.1.8.10 MCTP UUID - Time Low LSB - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	MCTP UUID - Time Low LSB	0xFFFF	

6.1.8.11 MCTP UUID - Time Low MSB - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	MCTP UUID - Time Low MSB	0xFFFF	

6.1.8.12 MCTP UUID - Time MID - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	MCTP UUID - Time MID	0xFFFF	

6.1.8.13 MCTP UUID - Time High and Version - 0x000C

Bits	Field Name	Default NVM Value	Description
15:0	MCTP UUID - Time High and Version	0xFFFF	

6.1.8.14 MCTP UUID - Clock Seq - 0x000D

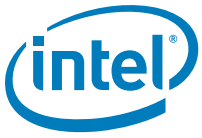
Bits	Field Name	Default NVM Value	Description
15:0	MCTP UUID - Clock Seq	0xFFFF	

6.1.8.15 MCTP UUID - Node1 - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	MCTP UUID - Node1	0x000000c9a000	

6.1.8.16 MCTP UUID - Node2 - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	MCTP UUID - Node2		



6.1.8.17 MCTP UUID - Node3 - 0x0010

Bits	Field Name	Default NVM Value	Description
15:0	MCTP UUID - Node3		

6.1.8.18 Alternative IANA - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	Alternative IANA	0x0	

6.1.8.19 NC_SI Over MCTP Message Types - 0x0012

Bits	Field Name	Default NVM Value	Description
15:8	NC-SI Command packet type	0x2	
7:0	NC-SI Pass Through packet type	0x3	

6.1.8.20 NC_SI Over MCTP Configuration - 0x0013

Bits	Field Name	Default NVM Value	Description
15:7	Reserved	0x0	
6	Simplified MCTP	0x0	0x0 Disabled 0x1 Enabled
5:1	Reserved	0x0	
0	Use Route By ID for Endpoint Discovery responses	0x0	0x0 Endpoint Discovery responses uses Route to Root Co 0x1 Endpoint Discovery responses uses Route by ID

6.1.8.21 MCTP Rate Limiter Configuration 1 - 0x0014

Bits	Field Name	Default NVM Value	Description
15:0	MCTP Rate	0x0	0x0 for no rate limitation count in sending MCTP's VDM messages

6.1.8.22 MCTP Rate Limiter Configuration 2 - 0x0015

Bits	Field Name	Default NVM Value	Description
15	Decision Point	0x0	0x0 - only full MCTP are being sent (no single VDM)
14:9	Reserved	0x0	
8:0	MCTP max credits	0xF0	

6.1.9 Flexible TCO Filter Configuration Section Summary Table

Word Offset	Description	Reference
0x0000	Section Header	See section 6.1.9.1
0x0001	Flexible Filter Length And Control	See section 6.1.9.2

6.1.9.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:8	Block CRC8		CRC-8-CCITT: Start Section -> Word: Flexible TCO Filter Configuration -> Section Header End Section -> Word: Flexible TCO Filter Configuration -> Flexible Filter Length And Control
7:0	Block Length		Length in words of the section covered by CRC

6.1.9.2 Flexible Filter Length and Control - 0x0001

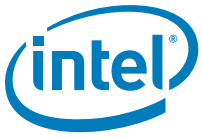
Bits	Field Name	Default NVM Value	Description
15:8	Flexible Filter Length (Bytes)	0x0	
7:5	Reserved5_7	0x0	
4	Last Filter	0x1	
3:2	Filter Index	0x0	
1	Reserved	0x0	
0	Apply Filter to LAN	0x0	0x0 Do Not Apply Flex Filter 0x1 Apply Flex Filter

6.1.10 Traffic Types Parameters Section Summary Table

Word Offset	Description	Reference
0x0000	Section Header	See section 6.1.10.1
0x0001	Traffic Type Data	See section 6.1.10.2

6.1.10.1 Section Header - 0x0000

Bits	Field Name	Default NVM Value	Description
15:8	Block CRC8		CRC-8-CCITT: Start Section -> Word: Traffic Types Parameters -> Section Header End Section -> Word: Traffic Types Parameters -> Traffic Type Data
7:0	Block Length		Length in words of the section covered by CRC



6.1.10.2 Traffic Type Data - 0x0001

Bits	Field Name	Default NVM Value	Description
15:2	Reserved	0x0	
1:0	traffic types	0x1	Valid values are: 0x0 Reserved 0x1 Network to BMC traffic only allowed 0x2 OS2BMC traffic only allowed 0x3 Both Network to BMC traffic and OS2BMC traffic all

6.1.11 PHY Configuration Section Summary Table

Word Offset	Description	Reference
0x0000	Section Length	See section 6.1.11.1
0x0001	Block CRC8	See section 6.1.11.2
0x0002	SGMII ADD 1	See section 6.1.11.3
0x0003	SGMII DATA 1	See section 6.1.11.4
0x0004	SGMII ADD 2	See section 6.1.11.5
0x0005	SGMII DATA 2	See section 6.1.11.6
0x0006	SGMII ADD 3	See section 6.1.11.7
0x0007	SGMII DATA 3	See section 6.1.11.8
0x0008	SGMII ADD 4	See section 6.1.11.9
0x0009	SGMII DATA 4	See section 6.1.11.10
0x000A	SGMII ADD 1 MDIO	See section 6.1.11.11
0x000B	SGMII DATA 1 MDIO	See section 6.1.11.12
0x000C	SGMII ADD 2 MDIO	See section 6.1.11.13
0x000D	SGMII DATA 2 MDIO	See section 6.1.11.14
0x000E	SGMII ADD 3 MDIO	See section 6.1.11.15
0x000F	SGMII DATA 3 MDIO	See section 6.1.11.16
0x0010	SGMII ADD 4 MDIO	See section 6.1.11.17
0x0011	SGMII DATA 4 MDIO	See section 6.1.11.18
0x0012	Serdes PHY Number & PHY Reg Address	See section 6.1.11.19
0x0013	Serdes PHY Reg Data	See section 6.1.11.20
0x0014	Serdes PHY Number & PHY Reg Address 1 reg 18	See section 6.1.11.21
0x0015	Serdes PHY 18_254 Reg Data	See section 6.1.11.22
0x0016	ADD	See section 6.1.11.23
0x0017	DATA	See section 6.1.11.24
0x0018	0_PHY Number & PHY Reg Address	See section 6.1.11.25
0x0019	0_PHY Reg Data	See section 6.1.11.26

6.1.11.1 Section Length - 0x0000

Bits	Field Name	Default NVM Value	Description
15	Reserved	0x0	
14:0	Section Length	0x4	Length in words of the section covered by CRC

6.1.11.2 Block CRC8 - 0x0001

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	
7:0	CRC8		CRC-8-CCITT: Start Section -> Word: PHY Configuration -> SGMII ADD 1 End Section -> Word: PHY Configuration -> 0_PHY Reg Data

6.1.11.3 SGMII ADD 1 - 0x0002

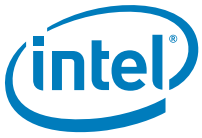
Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x2	
13:9	Reserved	0x0	
8	Loading Event	0x1	
7:0	PHY Register Address	0x1b	

6.1.11.4 SGMII DATA 1 - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	SGMII DATA 1	0x8480	

6.1.11.5 SGMII ADD 2 - 0x0004

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x2	Valid values are: 0x0 Internal PHY via MDIO Internal PHY, via MDIC register 0x1 External PHY via MDIO External PHY over I2C pins, via MDIC register 0x2 External PHY via I2C External PHY over I2C pins, via I2CCMD register 0x3 External via SDP External PHY over SDP pins, via I2CCMD register
13:9	Reserved	0x0	
8	Loading Event	0x1	0x0 PHY reset PHY resets (including POR) 0x1 Power on reset (SGMII) POR only
7:0	PHY Register Address	0x0	



6.1.11.6 SGMII DATA 2 - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	SGMII DATA 2	0x4091	

6.1.11.7 SGMII ADD 3 - 0x0006

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x2	
13:9	Reserved	0x0	
8	Loading Event	0x1	
7:0	PHY Register Address	0x09	

6.1.11.8 SGMII DATA 3 - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	SGMII DATA 3	0xf	

6.1.11.9 SGMII ADD 4 - 0x0008

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x2	Valid values are: 0x0 Internal PHY via MDIO Internal PHY, via MDIC register 0x1 External PHY via MDIO External PHY over I2C pins, via MDIC register 0x2 External PHY via I2C External PHY over I2C pins, via I2CCMD register 0x3 External via SDP External PHY over SDP pins, via I2CCMD register
13:9	Reserved	0x0	
8	Loading Event	0x1	0x0 PHY reset PHY resets (including POR) 0x1 Power on reset (SGMII) POR only
7:0	PHY Register Address	0x0	

6.1.11.10 SGMII DATA 4 - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	SGMII DATA 4	0x4096	

6.1.11.11 SGMII ADD 1 MDIO - 0x000A

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x1	

Bits	Field Name	Default NVM Value	Description
13:9	Reserved	0x0	
8	Loading Event	0x1	
7:0	PHY Register Address	0x1b	

6.1.11.12 SGMII DATA 1 MDIO - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	SGMII DATA 1	0x8084	

6.1.11.13 SGMII ADD 2 MDIO - 0x000C

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x1	
13:9	Reserved	0x0	
8	Loading Event	0x1	
7:0	PHY Register Address	0x0	

6.1.11.14 SGMII DATA 2 MDIO - 0x000D

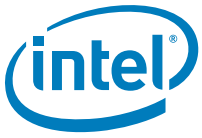
Bits	Field Name	Default NVM Value	Description
15:0	SGMII DATA 2	0x9140	

6.1.11.15 SGMII ADD 3 MDIO - 0x000E

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x1	
13:9	Reserved	0x0	
8	Loading Event	0x1	
7:0	PHY Register Address	0x09	

6.1.11.16 SGMII DATA 3 MDIO - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	SGMII DATA 3	0x0F00	



6.1.11.17 SGMII ADD 4 MDIO - 0x0010

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x1	
13:9	Reserved	0x0	
8	Loading Event	0x1	
7:0	PHY Register Address	0x0	

6.1.11.18 SGMII DATA 4 MDIO - 0x0011

Bits	Field Name	Default NVM Value	Description
15:0	SGMII DATA 4	0x9640	

6.1.11.19 Serdes PHY Number & PHY Reg Address - 0x0012

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x0	Valid values are: 0x0 Internal PHY via MDIO Internal PHY, via MDIC register 0x1 External PHY via MDIO External PHY over I2C pins, via MDIC register 0x2 External PHY via I2C External PHY over I2C pins, via I2CCMD register 0x3 External via SDP External PHY over SDP pins, via I2CCMD register
13:9	Reserved	0x0	
8	Loading Event	0x0	0x0 PHY reset PHY resets (including POR) 0x1 Power on reset (SGMII) POR only
7:0	PHY Reg Address	0x16	

6.1.11.20 Serdes PHY Reg Data - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	data	0xfe	

6.1.11.21 Serdes PHY Number & PHY Reg Address 1 reg 18 - 0x0014

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x0	Valid values are: 0x0 Internal PHY via MDIO Internal PHY, via MDIC register 0x1 External PHY via MDIO External PHY over I2C pins, via MDIC register 0x2 External PHY via I2C External PHY over I2C pins, via I2CCMD register 0x3 External via SDP External PHY over SDP pins, via I2CCMD register
13:9	Reserved	0x0	
8	Loading Event	0x0	0x0 PHY reset PHY resets (including POR) 0x1 Power on reset (SGMII) POR only

Bits	Field Name	Default NVM Value	Description
7:0	PHY Reg Address	0x12	

6.1.11.22 Serdes PHY 18_254 Reg Data - 0x0015

Bits	Field Name	Default NVM Value	Description
15:6	reserved	0x42	
5:4	18_254 data	0x2	
3:0	reserved	0xc	

6.1.11.23 ADD - 0x0016

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x2	Valid values are: 0x0 Internal PHY via MDIO Internal PHY, via MDIC register 0x1 External PHY via MDIO External PHY over I2C pins, via MDIC register 0x2 External PHY via I2C External PHY over I2C pins, via I2CCMD register 0x3 External via SDP External PHY over SDP pins, via I2CCMD register
13:9	Reserved	0x0	
8	Loading Event	0x1	0x0 PHY reset PHY resets (including POR) 0x1 Power on reset (SGMII) POR only
7:0	PHY Register Address	0x0	

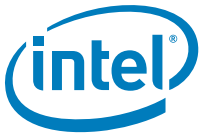
6.1.11.24 DATA - 0x0017

Bits	Field Name	Default NVM Value	Description
15:0	DATA	0x0	

6.1.11.25 0_PHY Number & PHY Reg Address - 0x0018

set page to 0

Bits	Field Name	Default NVM Value	Description
15:14	Transaction Interface	0x0	Valid values are: 0x0 Internal PHY via MDIO Internal PHY, via MDIC register 0x1 External PHY via MDIO External PHY over I2C pins, via MDIC register 0x2 External PHY via I2C External PHY over I2C pins, via I2CCMD register 0x3 External via SDP External PHY over SDP pins, via I2CCMD register
13:9	Reserved	0x0	
8	Loading Event	0x0	Valid values are: 0x0 PHY reset PHY resets (including POR) 0x1 Power on reset (SGMII) POR only
7:0	PHY Reg Address	0x16	



6.1.11.26 0_PHY Reg Data - 0x0019

Bits	Field Name	Default NVM Value	Description
15:0	data	0x0	

6.1.12 Alternate MAC Address Section Summary Table

Word Offset	Description	Reference
0x0000	Alternate MAC Address 0 - Word 0	See section 6.1.12.1
0x0001	Alternate MAC Address 0 - Word 1	See section 6.1.12.2
0x0002	Alternate MAC Address 0 - Word 2	See section 6.1.12.3
0x0003	Alternate MAC Address 1 - Word 0	See section 6.1.12.4
0x0004	Alternate MAC Address 1 - Word 1	See section 6.1.12.5
0x0005	Alternate MAC Address 1 - Word 2	See section 6.1.12.6
0x0006	Alternate MAC Address 2 - Word 0	See section 6.1.12.7
0x0007	Alternate MAC Address 2 - Word 1	See section 6.1.12.8
0x0008	Alternate MAC Address 2 - Word 2	See section 6.1.12.9
0x0009	Alternate MAC Address 3 - Word 0	See section 6.1.12.10
0x000A	Alternate MAC Address 3 - Word 1	See section 6.1.12.11
0x000B	Alternate MAC Address 3 - Word 2	See section 6.1.12.12

6.1.12.1 Alternate MAC Address 0 - Word 0 - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 0 - Word 0	0xFFFF	

6.1.12.2 Alternate MAC Address 0 - Word 1 - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 0 - Word 1	0xFFFF	

6.1.12.3 Alternate MAC Address 0 - Word 2 - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 0 - Word 2	0xFFFF	

6.1.12.4 Alternate MAC Address 1 - Word 0 - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 1 - Word 0	0xFFFF	

6.1.12.5 Alternate MAC Address 1 - Word 1 - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 1 - Word 1	0xFFFF	

6.1.12.6 Alternate MAC Address 1 - Word 2 - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 1 - Word 2	0xFFFF	

6.1.12.7 Alternate MAC Address 2 - Word 0 - 0x0006

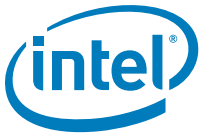
Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 2 - Word 0	0xFFFF	

6.1.12.8 Alternate MAC Address 2 - Word 1 - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 2 - Word 1	0xFFFF	

6.1.12.9 Alternate MAC Address 2 - Word 2 - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 2 - Word 2	0xFFFF	



6.1.12.10 Alternate MAC Address 3 - Word 0 - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 3 - Word 0	0xFFFF	

6.1.12.11 Alternate MAC Address 3 - Word 1 - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 3 - Word 1	0xFFFF	

6.1.12.12 Alternate MAC Address 3 - Word 2 - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	Alternate MAC Address 3 - Word 2	0xFFFF	

6.1.13 PBA Section Summary Table

Word Offset	Description	Reference
0x0000	PBA Section Length	See section 6.1.13.1
0x0001	Word1	See section 6.1.13.2
0x0002	Word2	See section 6.1.13.3
0x0003	Word3	See section 6.1.13.4
0x0004	Word4	See section 6.1.13.5
0x0005	Word5	See section 6.1.13.6

6.1.13.1 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

6.1.13.2 Word1 - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Word1 field		PBA Number stored in hexadecimal ASCII values.

6.1.13.3 Word2 - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Word2 field		PBA Number stored in hexadecimal ASCII values.

6.1.13.4 Word3 - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	Word3 field		PBA Number stored in hexadecimal ASCII values.

6.1.13.5 Word4 - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	Word4 field		PBA Number stored in hexadecimal ASCII values.

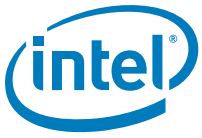
6.1.13.6 Word5 - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	Word5 field		PBA Number stored in hexadecimal ASCII values.

6.1.14 iSCSI Boot Configuration Section Summary Table

900 bytes, ptr is RO

Word Offset	Description	Reference
0x0430	iSCSI Boot Signature	See section 6.1.14.1
0x0431	iSCSI Block Size	See section 6.1.14.2
0x0432	Structure Version	See section 6.1.14.3
0x0433 + 1*n, n=0...127	Initiator Name	See section 6.1.14.4
0x04B3 + 1*n, n=0...16	Reserved1	See section 6.1.14.5
0x04C4	Flags	See section 6.1.14.6
0x04C5 + 1*n, n=0...1	Initiator IP	See section 6.1.14.7
0x04C7 + 1*n, n=0...1	Subnet Mask	See section 6.1.14.8
0x04C9 + 1*n, n=0...1	Gateway IP	See section 6.1.14.9
0x04CB	Boot LUN	See section 6.1.14.10
0x04CC + 1*n, n=0...1	Target IP	See section 6.1.14.11



Word Offset	Description	Reference
0x04CE	Target Port	See section 6.1.14.12
0x04CF + 1*n, n=0...127	Target Name	See section 6.1.14.13
0x054F + 1*n, n=0...8	Chap password	See section 6.1.14.14
0x0558 + 1*n, n=0...63	CHAP User Name	See section 6.1.14.15
0x0598	VLAN ID	See section 6.1.14.16
0x0599 + 1*n, n=0...8	mutual CHAP secret	See section 6.1.14.17
0x05A2 + 1*n, n=0...79	Reserves3	See section 6.1.14.18

6.1.14.1 iSCSI Boot Signature - 0x0430

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Boot Signature	0x5369	

6.1.14.2 iSCSI Block Size - 0x0431

Total byte size of the iSCSI configuration block

Bits	Field Name	Default NVM Value	Description
15:0	iSCSI Block Size	0x0384	Total byte size of the iSCSI configuration block

6.1.14.3 Structure Version - 0x0432

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x00	
7:0	Structure Version	0x01	Version number of iSCSI structure

6.1.14.4 Initiator Name[n] (0x0433 + 1*n, n=0...127)

iSCSI initiator name. This field is optional and built by manual input, DHCP host name, or with MAC address

Bits	Field Name	Default NVM Value	Description
15:0	Initiator Name	0x00	

6.1.14.5 Reserved1[n] (0x04B3 + 1*n, n=0...16)

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x00	

6.1.14.6 Flags - 0x04C4

Bits	Field Name	Default NVM Value	Description
15:10	ARP Timeout	0x0f	Connection timeout
9:8	ARP Retries	0x1	Retry value
7:6	Reserved	0x0	
5:4	Ctrl-D setup menu	0x0	Valid values are: 0x0 enabled 0x3 disabled, skip Ctrl-D entry
3:2	Authentication Type	0x0	Valid values are: 0x0 none 0x1 one way chap 0x2 mutual chap
1	Enable DHCP for getting iSCSI target information	0x1	Valid values are: 0x0 Use static target configuration 0x1 Use DHCP to get target information by the Option 1
0	Enable DHCP	0x1	Valid values are: 0x0 Use static configurations from this structure 0x1 Overrides configurations retrieved from DHCP

6.1.14.7 Initiator IP[n] (0x04C5 + 1*n, n=0...1)

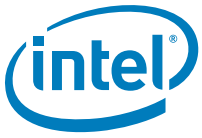
Bits	Field Name	Default NVM Value	Description
15:0	Initiator IP	0x0	

6.1.14.8 Subnet Mask[n] (0x04C7 + 1*n, n=0...1)

Bits	Field Name	Default NVM Value	Description
15:0	Subnet Mask	0x0	

6.1.14.9 Gateway IP[n] (0x04C9 + 1*n, n=0...1)

Bits	Field Name	Default NVM Value	Description
15:0	Gateway IP	0x0	



6.1.14.10 Boot LUN - 0x04CB

Bits	Field Name	Default NVM Value	Description
15:0	Boot LUN	0x0	

6.1.14.11 Target IP[n] (0x04CC + 1*n, n=0...1)

Bits	Field Name	Default NVM Value	Description
15:0	Target IP	0x0	

6.1.14.12 Target Port - 0x04CE

Bits	Field Name	Default NVM Value	Description
15:0	Target Port	0x0cbc	

6.1.14.13 Target Name[n] (0x04CF + 1*n, n=0...127)

Bits	Field Name	Default NVM Value	Description
15:0	Target Name	0x0	

6.1.14.14 Chap Password[n] (0x054F + 1*n, n=0...8)

Bits	Field Name	Default NVM Value	Description
15:0	Chap password	0x0	

6.1.14.15 CHAP User Name[n] (0x0558 + 1*n, n=0...63)

Bits	Field Name	Default NVM Value	Description
15:0	CHAP User Name	0x0	

6.1.14.16 VLAN ID - 0x0598

Bits	Field Name	Default NVM Value	Description
15:0	VLAN ID	0x0	

6.1.14.17 Mutual CHAP Secret [n] (0x0599 + 1*n, n=0...8)

Bits	Field Name	Default NVM Value	Description
15:0	mutual CHAP secret	0x0	

6.1.14.18 Reserves3[n] (0x05A2 + 1*n, n=0...79)

Bits	Field Name	Default NVM Value	Description
15:0	Reserves3	0x0	

6.1.15 VPD Module Section Summary Table

6.1.16 Pointer - GPHY FW Section Summary Table

Pointer to GPHY Arc Code

Word Offset	Description	Reference
0x07F0	GPHY FW Area Pointer	See section 6.1.16.1

6.1.16.1 GPHY FW Area Pointer - 0x07F0

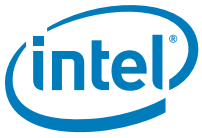
Bits	Field Name	Default NVM Value	Description
15	4K Ptr type mark	0x1	
14:0	GPHY FW Area Pointer	0x0	

Field GPHY SW Area Pointer points to GPHY SW Section. For GPHY SW inner structure See section 6.1.19

6.1.17 Pointer - SW Free Space Section Summary Table

Pointer to SW Free Space (leave 4 words empty before the sector end)

Word Offset	Description	Reference
0x07F1	SW Free Space Pointer	See section 6.1.17.1



6.1.17.1 SW Free Space Pointer - 0x07F1

Bits	Field Name	Default NVM Value	Description
15	4K Ptr type mark	0x1	
14:0	SW Free Space Pointer	0x0	

Field SW Free Space Pointer points to SW Free Space Section. For SW Free Space inner structure See section 6.1.28

6.1.18 EXP. ROM Boot Code Section Summary Table

512 KB - Expansion/Option ROM Module

6.1.19 GPHY SW Section Summary Table

128 KB - CHD GPHY ARC Code

6.1.20 CSS header Section Summary Table

CSS Header of FW Secured Module

Word Offset	Description	Reference
0x0000	moduleTypeL	See section 6.1.20.1
0x0001	moduleTypeH	See section 6.1.20.2
0x0002	headerLenL	See section 6.1.20.3
0x0003	headerLenH	See section 6.1.20.4
0x0004	headerVersionL	See section 6.1.20.5
0x0005	headerVersionH	See section 6.1.20.6
0x0006	moduleIDL	See section 6.1.20.7
0x0007	moduleIDH	See section 6.1.20.8
0x0008	moduleVendorL	See section 6.1.20.9
0x0009	moduleVendorH	See section 6.1.20.10

6.1.20.1 moduleTypeL - 0x0000

0x06 is Generic Module type

Bits	Field Name	Default NVM Value	Description
15:0	moduleType	0x6	

6.1.20.2 moduleTypeH - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	moduleTypeH		

6.1.20.3 headerLenL - 0x0002

0xA1 (161 [Decimal]) = HeaderDWs(32) + ModulusDWs*2 + ExponentDW(1)

Bits	Field Name	Default NVM Value	Description
15:0	headerLenL	0xa1	

6.1.20.4 headerLenH - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	headerLenH		

6.1.20.5 headerVersionL - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	headerVersionL	0x00010000	

6.1.20.6 headerVersionH - 0x0005

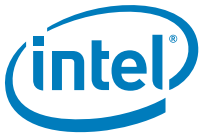
Bits	Field Name	Default NVM Value	Description
15:0	headerVersionH		

6.1.20.7 moduleIDL - 0x0006

Bits	Field Name	Default NVM Value	Description
15:0	moduleIDL	0x0	

6.1.20.8 moduleIDH - 0x0007

Bits	Field Name	Default NVM Value	Description
15	signMode	0x1	MSB indicates Debug/Release : 1 = Debug 0 = Release
14:0	moduleIDH		



6.1.20.9 moduleVendorL - 0x0008

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorL	0x8086	

6.1.20.10 moduleVendorH - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	moduleVendorH		

6.1.21 FW_HEADER in REGMAN Section Summary Table

Word Offset	Description	Reference
0x0000	DateL	See section 6.1.21.1
0x0001	DateH	See section 6.1.21.2
0x0002	*sizeL	See section 6.1.21.3
0x0003	sizeH	See section 6.1.21.4
0x0004	keySizeL	See section 6.1.21.5
0x0005	keySizeH	See section 6.1.21.6
0x0006	modulusSizeL	See section 6.1.21.7
0x0007	modulusSizeH	See section 6.1.21.8
0x0008	exponentSizeL	See section 6.1.21.9
0x0009	exponentSizeH	See section 6.1.21.10
0x000A	lad_srevL	See section 6.1.21.11
0x000B	lad_srevH	See section 6.1.21.12
0x000C	reserved1	See section 6.1.21.13
0x000D	reserved2	See section 6.1.21.14
0x000E	lad_fw_entry_offsetL	See section 6.1.21.15
0x000F	lad_fw_entry_offsetH	See section 6.1.21.16
0x0010	lad_fl_dev_offsetL	See section 6.1.21.17
0x0011	lad_fl_dev_offsetH	See section 6.1.21.18
0x0012 + 1*n, n=0...35	reserved	See section 6.1.21.19
0x0036 + 1*n, n=0...127	RSA Public Key	See section 6.1.21.20
0x00B6	RSA ExponentL	See section 6.1.21.21
0x00B7	RSA ExponentH	See section 6.1.21.22
0x00B8 + 1*n, n=0...127	Encrypted SHA256 Hash	See section 6.1.21.23

6.1.21.1 DateL - 0x0000

yyyymmdd BCD

Bits	Field Name	Default NVM Value	Description
15:0	DateL	0x20170820	

6.1.21.2 DateH - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	DateH		

6.1.21.3 *sizeL - 0x0002

Size of all secured area (CSS) - DWORD size of whole module (headers and code)

Bits	Field Name	Default NVM Value	Description
15:0	sizeL	0x3ec00	

6.1.21.4 sizeH - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	sizeH		

6.1.21.5 keySizeL - 0x0004

DWORD size of RSA modulus (2048 or 3072)/32

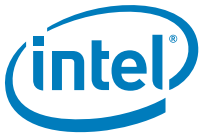
Bits	Field Name	Default NVM Value	Description
15:0	keySizeL	0x40	

6.1.21.6 keySizeH - 0x0005

Bits	Field Name	Default NVM Value	Description
15:0	keySizeH		

6.1.21.7 modulusSizeL - 0x0006

DWORD size of RSA modulus (2048 or 3072)/32



Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeL	0x40	

6.1.21.8 modulusSizeH - 0x0007

Bits	Field Name	Default NVM Value	Description
15:0	modulusSizeH		

6.1.21.9 exponentSizeL - 0x0008

DWORD size of RSA exponent (1 DW)

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeL	0x1	

6.1.21.10 exponentSizeH - 0x0009

Bits	Field Name	Default NVM Value	Description
15:0	exponentSizeH		

6.1.21.11 lad_srevL - 0x000A

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevL	0x1	

6.1.21.12 lad_srevH - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	lad_srevH		

6.1.21.13 reserved1 - 0x000C

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.1.21.14 reserved2 - 0x000D

Bits	Field Name	Default NVM Value	Description
15:0	lad_nvmsizeH		

6.1.21.15 lad_fw_entry_offsetL - 0x000E

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetL	0x14c	

6.1.21.16 lad_fw_entry_offsetH - 0x000F

Bits	Field Name	Default NVM Value	Description
15:0	lad_fw_entry_offsetH		

6.1.21.17 lad_fl_dev_offsetL - 0x0010

Bits	Field Name	Default NVM Value	Description
15:0	lad_fl_dev_offsetL	0x2c800	

6.1.21.18 lad_fl_dev_offsetH - 0x0011

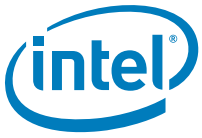
Bits	Field Name	Default NVM Value	Description
15:0	lad_fl_dev_offsetH		

6.1.21.19 reserved[n] (0x0012 + 1*n, n=0...35)

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0x0	

6.1.21.20 RSA Public Key [n] (0x0036 + 1*n, n=0...127)

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	



6.1.21.21 RSA ExponentL - 0x00B6

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentL	0x0	

6.1.21.22 RSA ExponentH - 0x00B7

Bits	Field Name	Default NVM Value	Description
15:0	RSA ExponentH	0x0	

6.1.21.23 Encrypted SHA256 Hash[n] (0x00B8 + 1*n, n=0...127)

Bits	Field Name	Default NVM Value	Description
15:0	RSA Public Key	0x0	

6.1.22 FW Image Section Summary Table

356 KB - MNG FW Secured Module

6.1.23 Flash Info Section Summary Table

4 KB - Trailer of FW Secured Module

Word Offset	Description	Reference
0x0000	Flash Devices Table Version	See section 6.1.23.1
0x0001	Blank NVM Device ID	See section 6.1.23.2
0x0002	Minimum FW Code Revision	See section 6.1.23.3
0x0003	Number of Flash Devices	See section 6.1.23.4
0x0004	JEDEC_ID_L 18	See section 6.1.23.5
0x0005	JEDEC_ID_H 18	See section 6.1.23.6
0x0006	FLASHMODE_L 18	See section 6.1.23.7
0x0007	FLASHMODE_H+unprotect 18	See section 6.1.23.8
0x0008	FLASHOP_L 18	See section 6.1.23.9
0x0009	FLASHOP_H 18	See section 6.1.23.10
0x000A	*FLASHTIME_L 18	See section 6.1.23.11
0x000B	FLASHTIME_H 18	See section 6.1.23.12
0x000C	JEDEC_ID_L 19	See section 6.1.23.13
0x000D	JEDEC_ID_H 19	See section 6.1.23.14
0x000E	FLASHMODE_L 19	See section 6.1.23.15
0x000F	FLASHMODE_H+unprotect 19	See section 6.1.23.16

Word Offset	Description	Reference
0x0010	FLASHOP_L 19	See section 6.1.23.17
0x0011	FLASHOP_H 19	See section 6.1.23.18
0x0012	*FLASHTIME_L 19	See section 6.1.23.19
0x0013	FLASHTIME_H 19	See section 6.1.23.20
0x0014	JEDEC_ID_L 20	See section 6.1.23.21
0x0015	JEDEC_ID_H 20	See section 6.1.23.22
0x0016	FLASHMODE_L 20	See section 6.1.23.23
0x0017	FLASHMODE_H+unprotect 20	See section 6.1.23.24
0x0018	FLASHOP_L 20	See section 6.1.23.25
0x0019	FLASHOP_H 20	See section 6.1.23.26
0x001A	*FLASHTIME_L 20	See section 6.1.23.27
0x001B	FLASHTIME_H 20	See section 6.1.23.28
0x001C	JEDEC_ID_L 62_5	See section 6.1.23.29
0x001D	JEDEC_ID_H 62_5	See section 6.1.23.30
0x001E	FLASHMODE_L 62_5	See section 6.1.23.31
0x001F	FLASHMODE_H+unprotect 62_5	See section 6.1.23.32
0x0020	FLASHOP_L 62_5	See section 6.1.23.33
0x0021	FLASHOP_H 62_5	See section 6.1.23.34
0x0022	*FLASHTIME_L 62_5	See section 6.1.23.35
0x0023	FLASHTIME_H 62_5	See section 6.1.23.36

6.1.23.1 Flash Devices Table Version - 0x0000

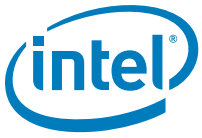
Bits	Field Name	Default NVM Value	Description
15:0	Flash Devices Table Version	0x0	

6.1.23.2 Blank NVM Device ID - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Device ID	0x15FD	

6.1.23.3 Minimum FW Code Revision - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Minimum FW Code Revision	0x0	



6.1.23.4 Number of Flash Devices - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	Number of Flash Devices	0x1	

6.1.23.5 JEDEC_ID_L 18 - 0x0004

Bits	Field Name	Default NVM Value	Description
15:0	JEDEC_ID_L	0x8d25BF	Winbond W25Q16DV-16Mbit

6.1.23.6 JEDEC_ID_H 18 - 0x0005

Bits	Field Name	Default NVM Value	Description
15:8	reserved	0x0	
7:0	JEDEC_ID_H		

6.1.23.7 FLASHMODE_L 18 - 0x0006

Bits	Field Name	Default NVM Value	Description
15:7	Reserved	0x0	
6	SST_MODE	0x1	
5	SUSPEND_SUPPORT	0x0	
4:3	FLASH_SPEED	0x2	
2:1	NUM_OF_DUMMY	0x1	
0	FAST_READ_MODE	0x1	

6.1.23.8 FLASHMODE_H+unprotect 18 - 0x0007

Bits	Field Name	Default NVM Value	Description
15	Unprotected flash after FW reset	0x1	
14:0	Reserved	0x0	

6.1.23.9 FLASHOP_L 18 - 0x0008

Bits	Field Name	Default NVM Value	Description
15:8	SUSPENDOP	0x75	
7:0	FLASHERASEOP	0xC7	

6.1.23.10 FLASHOP_H 18 - 0x0009

Bits	Field Name	Default NVM Value	Description
15:8	FASTREADOP	0x0B	
7:0	RESUMEOP	0x7A	

6.1.23.11 *FLASHTIME_L 18 - 0x000A

Bits	Field Name	Default NVM Value	Description
15:6	Reserved	0x0	
5:0	CSDESELECT	0x1F	Valid values are: 0x10 57.2 ns 0x15 73.6 ns 0x1A 89.6 ns 0x1F 105.6 ns (The time is measured in cycles of 3.4 ns plus 2 clock cycles)

6.1.23.12 FLASHTIME_H 18 - 0x000B

Bits	Field Name	Default NVM Value	Description
15:0	HOLDTIME	0x280	The time is measured in cycles of 6.4 ns. The default is 4 µs.

6.1.23.13 JEDEC_ID_L 19 - 0x000C

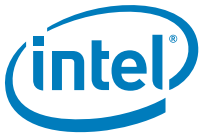
Bits	Field Name	Default NVM Value	Description
15:0	JEDEC_ID_L	0x008e25bf	Microchip SST25VF080B-8 Mbit

6.1.23.14 JEDEC_ID_H 19 - 0x000D

Bits	Field Name	Default NVM Value	Description
15:8	reserved	0x0	
7:0	JEDEC_ID_H		

6.1.23.15 FLASHMODE_L 19 - 0x000E

Bits	Field Name	Default NVM Value	Description
15:7	Reserved	0x0	
6	SST_MODE	0x1	
5	SUSPEND_SUPPORT	0x0	
4:3	FLASH_SPEED	0x2	



Bits	Field Name	Default NVM Value	Description
2:1	NUM_OF_DUMMY	0x1	
0	FAST_READ_MODE	0x1	

6.1.23.16 FLASHMODE_H+unprotect 19 - 0x000F

Bits	Field Name	Default NVM Value	Description
15	Unprotected flash after FW reset	0x1	
14:0	Reserved	0x0	

6.1.23.17 FLASHOP_L 19 - 0x0010

Bits	Field Name	Default NVM Value	Description
15:8	SUSPENDOP	0x75	
7:0	FLASHERASEOP	0xC7	

6.1.23.18 FLASHOP_H 19 - 0x0011

Bits	Field Name	Default NVM Value	Description
15:8	FASTREADOP	0x0B	
7:0	RESUMEOP	0x7A	

6.1.23.19 *FLASHTIME_L 19 - 0x0012

Bits	Field Name	Default NVM Value	Description
15:6	Reserved	0x0	
5:0	CSDESELECT	0x1F	Valid values are: 0x10 57.2 ns 0x15 73.6 ns 0x1A 89.6 ns 0x1F 105.6 ns (The time is measured in cycles of 3.4 ns plus 2 clock cycles)

6.1.23.20 FLASHTIME_H 19 - 0x0013

Bits	Field Name	Default NVM Value	Description
15:0	HOLDTIME	0x280	The time is measured in cycles of 6.4 ns. The default is 4 µs.

6.1.23.21 JEDEC_ID_L 20 - 0x0014

Bits	Field Name	Default NVM Value	Description
15:0	JEDEC_ID_L	0x4a25bf	Microchip SST25VF080B-8 Mbit

6.1.23.22 JEDEC_ID_H 20 - 0x0015

Bits	Field Name	Default NVM Value	Description
15:8	reserved	0x0	
7:0	JEDEC_ID_H		

6.1.23.23 FLASHMODE_L 20 - 0x0016

Bits	Field Name	Default NVM Value	Description
15:7	Reserved	0x0	
6	SST_MODE	0x1	
5	SUSPEND_SUPPORT	0x0	
4:3	FLASH_SPEED	0x2	
2:1	NUM_OF_DUMMY	0x1	
0	FAST_READ_MODE	0x1	

6.1.23.24 FLASHMODE_H+unprotect 20 - 0x0017

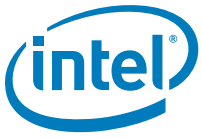
Bits	Field Name	Default NVM Value	Description
15	Unprotected flash after FW reset	0x1	
14:0	Reserved	0x0	

6.1.23.25 FLASHOP_L 20 - 0x0018

Bits	Field Name	Default NVM Value	Description
15:8	SUSPENDOP	0x75	
7:0	FLASHERASEOP	0xC7	

6.1.23.26 FLASHOP_H 20 - 0x0019

Bits	Field Name	Default NVM Value	Description
15:8	FASTREADOP	0x0B	
7:0	RESUMEOP	0x7A	



6.1.23.27 *FLASHTIME_L 20 - 0x001A

Bits	Field Name	Default NVM Value	Description
15:6	Reserved	0x0	
5:0	CSDESELECT	0x1F	Valid values are: 0x10 57.2 ns 0x15 73.6 ns 0x1A 89.6 ns 0x1F 105.6 ns (The time is measured in cycles of 3.4 ns plus 2 clock cycles)

6.1.23.28 FLASHTIME_H 20 - 0x001B

Bits	Field Name	Default NVM Value	Description
15:0	HOLDTIME	0x280	The time is measured in cycles of 6.4 ns. The default is 4 µs.

6.1.23.29 JEDEC_ID_L 62_5 - 0x001C

Bits	Field Name	Default NVM Value	Description
15:0	JEDEC_ID	0xffff	DEFUALT_62_5

6.1.23.30 JEDEC_ID_H 62_5 - 0x001D

Bits	Field Name	Default NVM Value	Description
15:8	Reserved	0x0	
7:0	JEDEC_ID_2	0xff	

6.1.23.31 FLASHMODE_L 62_5 - 0x001E

Bits	Field Name	Default NVM Value	Description
15:7	Reserved	0x0	
6	SST_MODE	0x0	
5	SUSPEND_SUPPORT	0x0	
4:3	FLASH_SPEED	0x02	
2:1	NUM_OF_DUMMY	0x01	
0	FAST_READ_MODE	0x01	

6.1.23.32 FLASHMODE_H+unprotect 62_5 - 0x001F

Bits	Field Name	Default NVM Value	Description
15	Unprotected flash	0x01	
14:0	FLASHMODE	0x0	

6.1.23.33 FLASHOP_L 62_5 - 0x0020

Bits	Field Name	Default NVM Value	Description
15:8	SUSPENDOP	0x75	
7:0	FLASHERASEOP	0xC7	

6.1.23.34 FLASHOP_H 62_5 - 0x0021

Bits	Field Name	Default NVM Value	Description
15:8	FASTREADOP	0x0B	
7:0	RESUMEOP	0x7A	

6.1.23.35 *FLASHTIME_L 62_5 - 0x0022

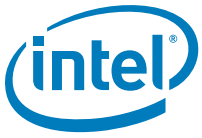
Bits	Field Name	Default NVM Value	Description
15:6	Reserved	0x0	
5:0	CSDESELECT	0x1F	Valid values are: 0x10 57.2 ns 0x15 73.6 ns 0x1A 89.6 ns 0x1F 105.6 ns (The time is measured in cycles of 3.4 ns plus 2 clock cycles)

6.1.23.36 FLASHTIME_H 62_5 - 0x0023

Bits	Field Name	Default NVM Value	Description
15:0	HOLDTIME	0x280	The time is measured in cycles of 6.4 ns. The default is 4 µs.

6.1.24 RO Updates Info Section Summary Table

4 KB - Trailer of FW Secured Module



Word Offset	Description	Reference
0x0000	RO Updates Version	See section 6.1.24.1
0x0001	Blank NVM Device ID	See section 6.1.24.2
0x0002	Minimum FW Code Revision	See section 6.1.24.3
0x0003	RO Updates Length	See section 6.1.24.4
0x0004	RO Updates Contents	See section 6.1.24.5

6.1.24.1 RO Updates Version - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	RO Commands Version	0xffff	

6.1.24.2 Blank NVM Device ID - 0x0001

Bits	Field Name	Default NVM Value	Description
15:0	Device ID	0x15FD	

6.1.24.3 Minimum FW Code Revision - 0x0002

Bits	Field Name	Default NVM Value	Description
15:0	Minimum FW Code Revision	0xffff	

6.1.24.4 RO Updates Length - 0x0003

Bits	Field Name	Default NVM Value	Description
15:0	RO Updates Length	0xffff	

6.1.24.5 RO Updates Contents - 0x0004

Raw data module length: variable

6.1.25 CSS-signed Firmware Secured Module Section Summary Table

CSS-Signed Secured FW (Secured Area)

6.1.26 Free Provisioning Area Section Summary Table

Size of all Secured Area - For double banking process (244KB+16KB - Free Provisioning Area+mDNS Records)

Word Offset	Description	Reference
0x0000	reserved	See section 6.1.26.1

6.1.26.1 Reserved - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	reserved	0xffff	

6.1.27 mDNS Records Section Summary Table

16 KB - mDNS Buffer

Word Offset	Description	Reference
0x0000	Reserved	See section 6.1.27.1

6.1.27.1 Reserved - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	Reserved	0x0000	

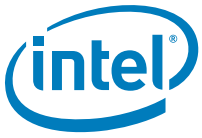
6.1.28 SW Free Space Section Summary Table

16 KB - Free Software Space

Word Offset	Description	Reference
0x0000	Reserved	See section 6.1.28.1

6.1.28.1 Reserved - 0x0000

Bits	Field Name	Default NVM Value	Description
15:0	reserved		



7.0 Inline Functions

7.1 Receive Functionality

Typically, packet reception consists of recognizing the presence of a packet on the wire, performing address filtering, storing the packet in the receive data FIFO, transferring the data to one of the 4 receive queues in host memory, and updating the state of a receive descriptor.

A received packet goes through two stages of filtering:

The first step is to verify that the packet is destined to the port. This is done by a set of L2 filters as described in Section 7.1.3.

In the second stage, a received packet that successfully passed the Rx filters is associated with one or more receive descriptor queues as described in Section 7.1.1.

7.1.1 L2 Packet Filtering MNG and Circuit Breaker filters

The receive packet filtering role is to determine which of the incoming packets are allowed to pass to the local system and which of the incoming packets should be dropped since they are not targeted to the local system. Received packets can be destined to the host, to a Manageability Controller (MC), or to both. This section describes how host filtering is done, and the interaction with management filtering.

As shown in Figure 7-1, host filtering has three stages:

1. Packets are filtered by the manageability and Circuit Breaker filters (IP, flex, other). See Section 10.4.1 for details on MNG filters and Section 10.4.2 for details on Circuit Breaker filters.
2. Packets are filtered by L2 filters (MAC address, unicast/multicast/broadcast). See Section 7.1.1.1 for details.
3. Packets are then filtered by VLAN if a VLAN tag is present. See Section 7.1.1.2 for details.

A packet is not forwarded to the host if any of the following takes place:

1. It is dropped or inhibited for the host by the MNG filters or the CB filters
2. The packet does not pass MAC address filters or the VLAN filters.

A packet that passes the receive filtering might still be dropped due to other reasons. Normally, only good packets are received. These are defined as those packets with no Under Size Error, Over Size Error (see Section 7.1.1.4), Packet Error, Length Error and CRC Error are detected. However, if the *store-bad-packet* bit is set (*RCTL.SBP*), then bad packets that pass the filter function are stored in host memory. Packet errors are indicated by error bits in the receive descriptor (*RDESC.ERRORS*). It is possible to receive all packets, regardless of whether they are bad, by setting the promiscuous enabled (Unicast and Multicast) and the *store-bad-packet* bits in the *RCTL* register.

If there is insufficient space in the receive FIFO, hardware drops the packet and indicates the missed packet in the appropriate statistics registers.

When the packet is routed to a queue with the *SRRCTL.Drop_En* bit set to 1b, receive packets are dropped when insufficient receive descriptors exist to write the packet into system memory.

Note: CRC errors before the SFD are ignored. Any packet must have a valid SFD in order to be recognized by Foxville (even bad packets).

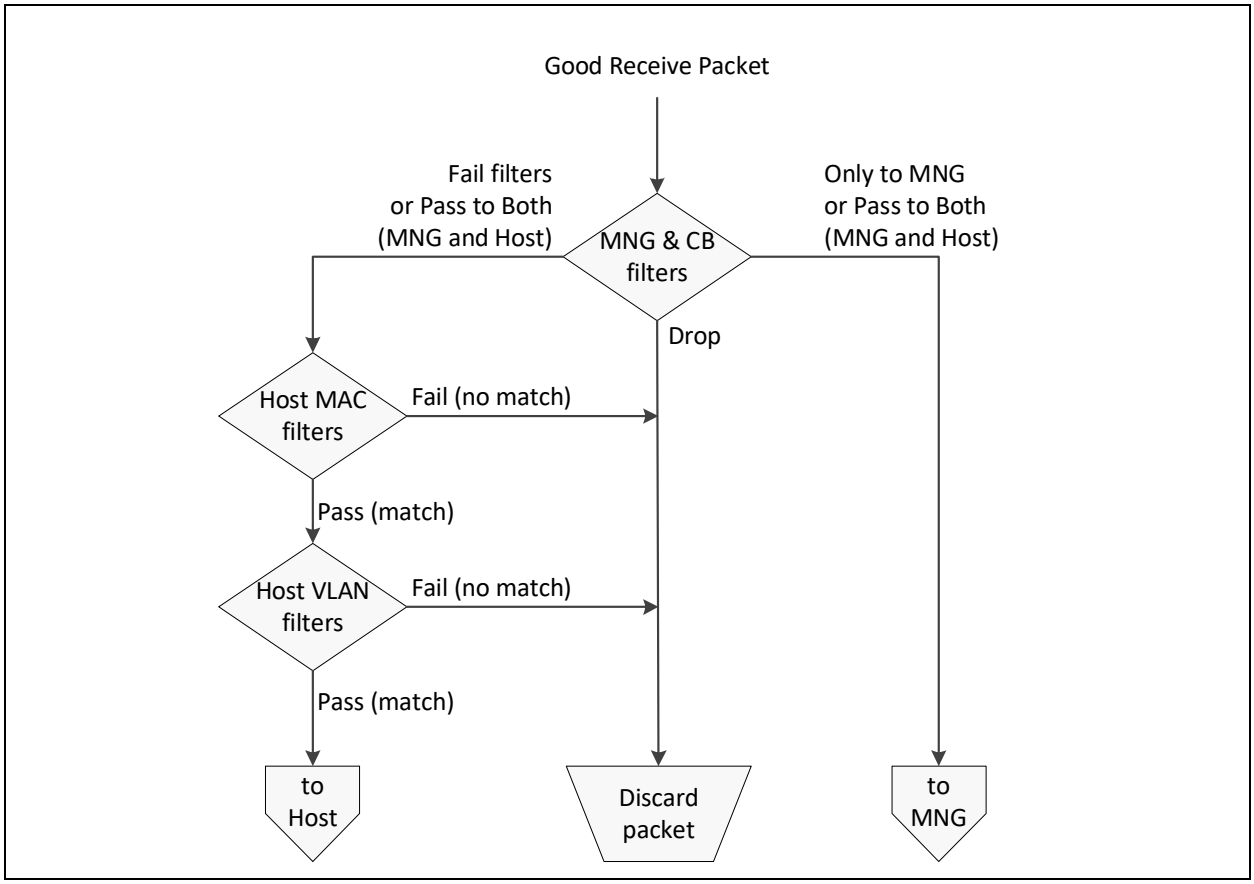


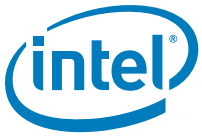
Figure 7-1. Foxville Receive Filtering Flow Chart

7.1.1.1 MAC Address Filtering

Figure 7-4 shows the MAC address filtering. A packet passes successfully through the MAC address filtering if any of the following conditions are met:

1. It is a unicast packet and promiscuous unicast filtering is enabled (RCTL.UPE).
2. It is a multicast packet and promiscuous multicast filtering is enabled (RCTL.UPE).
3. It is a broadcast packet and Broadcast Accept Mode (*RCTL.BAM*) is enabled.
4. It is a unicast or multicast packet and it matches one of the unique MAC filters (RAL/RAH 0...15).
5. It is a unicast or multicast packet and it matches the inexact based filters (MTA). The MTA can be enabled for multicast packets, unicast packets or both according to the RCTL.HSEL setting.

7.1.1.1.1 Unique (Exact) MAC Address Filters



The entire MAC address is checked against one of the unique MAC address filters listed above. The RAL/RAH registers can be set to check the source or destination MAC addresses while all the other filters check only the destination MAC address. The (selected) MAC address of an incoming packet must exactly match one of these filters.

7.1.1.1.2 Inexact MAC Address Filter

This filter is aimed for unicast packets, multicast packets or both, as selected by the RCTL.HSEL field. A 12-bit portion of the received MAC address must match the Multicast Filter Address table (MFA) in order to pass the filter. This means that 12 bits out of 48 bits of the destination address are used for the filtering. There are 4 options selecting which 12 bits are used for the filtering according to the RCTL.*MO* field. The 12 bits extracted from the Destination address are used as an address for a bit in the Multicast / Unicast Table Array (MTA). If the value of the bit selected in the MTA table is 1b, the packet passes the filter. Packets that match the MTA and do not match other MAC address filters are indicated by the *PIF* bit in the receive descriptor (PIF = Pass Inexact Filter). The software is expected to check these incoming packets against a list of permitted MAC addresses.

7.1.1.2 VLAN Filtering

A receive packet that successfully passed MAC address filtering is then subjected to VLAN header filtering.

1. If the packet does not have a VLAN header, it passes to the next filtering stage.

Note: If external VLAN is enabled (*CTRL_EXT.EXT_VLAN* is set), it is assumed that the first VLAN tag is an external VLAN and it is skipped. All next stages refer to the second VLAN.

2. If VLAN filtering is disabled (*RCTL.VFE* bit is cleared), the packet is forwarded to the next filtering stage.
3. If the packet has a VLAN header, and it matches an enabled host VLAN filter (relevant bit in VFTA table is set), the packet is forwarded to the next filtering stage.
4. Otherwise, the packet is dropped.

Figure 7-2 shows the VLAN filtering flow.

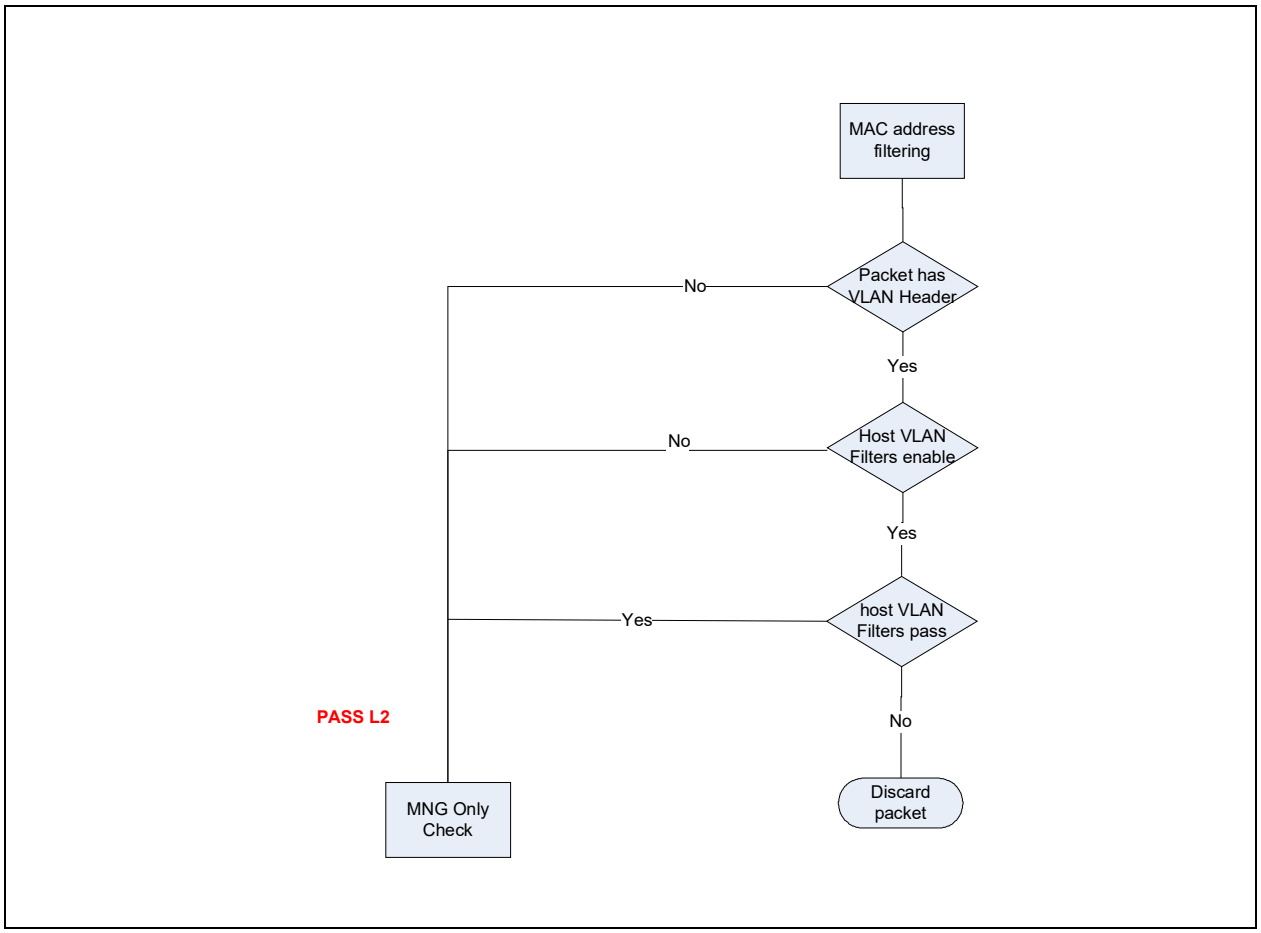


Figure 7-2. Foxville VLAN Filtering

7.1.1.3 Manageability Filtering

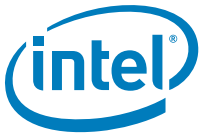
Manageability filtering is described in Section 10.4.

7.1.1.4 Size Filtering

A packet is defined as undersize if it is smaller than 64 bytes.

A packet is defined as oversize in the following conditions:

- The *RCTL.LPE* bit cleared and one of the following conditions is met:
  - The packet is bigger than 1518 bytes and there are no VLAN tags in the packet.
  - The packet is bigger than 1522 bytes and there is one VLAN tag in the packet.
  - The packet is bigger than 1526 bytes and there are two VLAN tags in the packet.
- The *RCTL.LPE* bit is set to 1b and the packet is bigger than *RLPML.RLPML* bytes.



Note: Even when the RCTL.LPE bit is set, the maximum supported received-packet size is 9.5 KB (9728 bytes).

7.1.2 Receive Queues Assignment

The following filter mechanisms determines the destination of a receive packet.

- RSS — Receive Side Scaling distributes packet processing between several processor cores by assigning packets into different descriptor queues. RSS assigns to each received packet an RSS index. Packets are routed to a queue out of a set of Rx queues based on their RSS index and other considerations. See Section 7.1.2.9 for details on RSS.
- L2 Ether-type filters — These filters identify packets by their L2 Ether-type and assign them to receive queues. Examples of possible uses are LLDP packets and 802.1X packets. See Section 7.1.2.3 for mode details. Foxville incorporates 4 Ether-type filters.
- 2-tuple filters — These filters identify packets with specific TCP/UDP destination port and/or L4 protocol. Each filter consists of a 2-tuple (protocol and destination TCP/UDP port) and routes packets into one of the Rx queues. Foxville has 8 such filters. See Section 7.1.2.4 for details.
- TCP SYN filters — Foxville might route TCP packets with their *SYN* flag set into a separate queue. *SYN* packets are often used in *SYN* attacks to load the system with numerous requests for new connections. By filtering such packets to a separate queue, security software can monitor and act on *SYN* attacks. Foxville has one such filter. See Section 7.1.2.6 for more details.
- Flex Filters - These filters can be either used as WoL filters when Foxville is in D3 state or for queueing in normal operating mode (D0 state). Filters enable queueing according to a match of any 128 Byte sequence at the beginning of a packet. Each one of the 128 bytes can be either compared or masked using a dedicated mask field. Foxville has 32 such filters. See Section 7.1.2.5 for details.
- VLAN priority filters — These filters identify packets by their L2 VLAN priority and assign them to receive queues. BCN packets, See Section 7.1.2.7 for mode details. Foxville incorporates 8 VLAN priority filters.
- MAC address filters — These filters identify packets by their L2 MAC address and assign them to receive queues. See Section 7.1.2.8 for mode details. Foxville incorporates 16 unique MAC address filters.

A received packet is allocated to a queue as described in the following sections.

The tables below describe allocation of queues in each of the modes.

Table 7-1. Queue Allocation

	RSS	Queue allocation
	Disabled	One default queue (*MRQC.DEF_Q*). Packets can be allocated to queues using special filters.
	Enabled	Up to 4 queues by RSS for packets not allocated by special filters.

7.1.2.1 Queuing Method

When the *MRQC.Multiple Receive Queues Enable* field equals 010b (multiple receive queues as defined by filters and RSS for 4 queues) or 000b (multiple receive queues as defined by filters (2-tuple filters, L2 Ether-type filters, SYN filter and Flex Filters), the received packet is assigned to a queue in the following manner (Each filter identifies one of 4 receive queues):

1. Queue by MAC address filters (if a match)
2. Queue by L2 Ether-type filters (if a match)
3. If RFCTL.SYNQFP is 0b (2-tuple filter and Flex filter have priority), then:

   a. Queue by Flex filter (if a match)
   b. Queue by 2-tuple filter
   c. Queue by SYN filter (if a match)
4. If RFCTL.SYNQFP is 1b (SYN filter has priority), then:
   a. Queue by SYN filter (if a match)
   b. Queue by Flex filter (if a match)
   c. Queue by 2-tuple filter (if a match)
5. Queue by VLAN Priority (if a match)
6. Queue by RSS (if RSS enabled) - Identifies one of 1 x 4 queues through the RSS index. The following modes are supported:
   — No RSS — The default queue as defined in *MRQC.DEF_Q* is used for packets that do not meet any of the previous conditions.
   — RSS only — A set of 4 queues is allocated for RSS. The queue is identified through the RSS index. Note that it is possible to use a subset of the 4 queues.

Note: No RSS here mean either that RSS is disabled (*MRQC.Multiple Receive Queues Enable* field equals 000b) or that the packet did not match any of the RSS filters.

Figure 7-3 shows receive queue assignment flow.

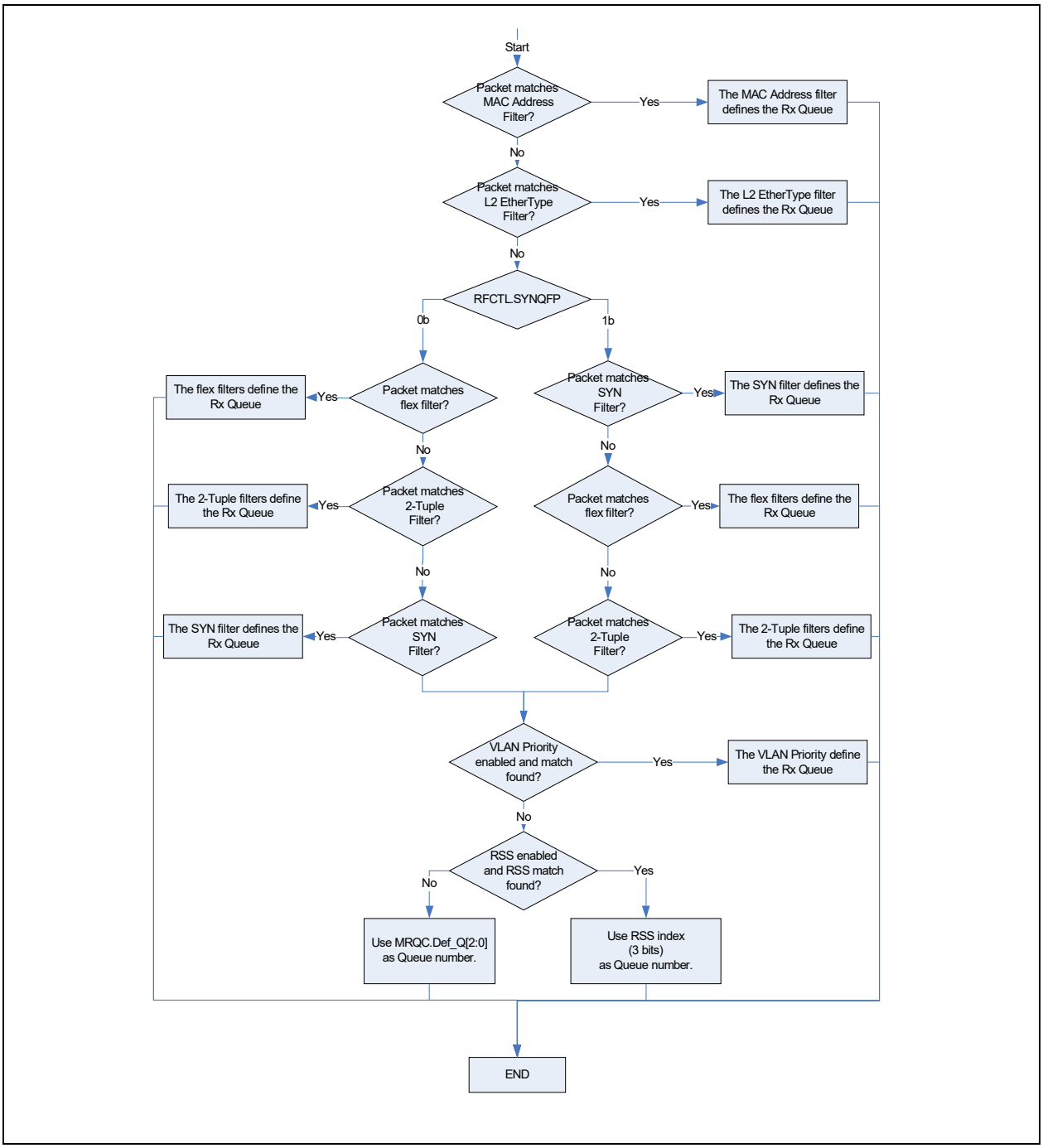


Figure 7-3. Receive Queuing Flow

7.1.2.2 Queue Configuration Registers

Configuration registers (CSRs) that control queue operation are replicated per queue (total of 4 copies of each register). Each of the replicated registers correspond to a queue such that the queue index equals the serial number of the register (such as register 0 corresponds to queue 0, etc.). Registers included in this category are:

- *RDBAL* and *RDBAH* — Rx Descriptor Base
- *RDLEN* — RX Descriptor Length
- *RDH* — RX Descriptor Head
- *RDT* — RX Descriptor Tail
- *RXDCTL* — Receive Descriptor Control
- *SRRCTL* — Split and Replication Receive Control
- *PSRTYPE* — Packet Split Receive Type

7.1.2.3 L2 Ether-type Filters

These filters identify packets by L2 Ether-type and assign them to a receive queue. The following usages have been identified:

- IEEE 802.1X packets — Extensible Authentication Protocol over LAN (EAPOL).
- Time sync packets (such as IEEE 1588) — Identifies Sync or Delay_Req packets
- IEEE802.1AB LLDP (Link Layer Discovery Protocol) packets.
- IEEE1722 (Layer 2 Transport Protocol for Time Sensitive Applications) packets
- IEEE1722 Layer 2 transport protocol for timed sensitive applications.

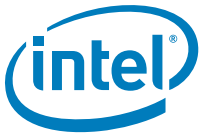
Foxville incorporates 8 Ether-type filters.

The *Packet Type* field in the Rx descriptor captures the filter number that matched the L2 Ether-type. See Section 7.1.4.2 for decoding of the *Packet Type* field.

The Ether-type filters are configured via the ETQF register as follows:

- The *EType* field contains the 16-bit Ether-type compared against all L2 type fields in the Rx packet.
- The *Filter Enable* bit enables identification of Rx packets by Ether-type according to this filter. If this bit is cleared, the filter is ignored for all purposes.
- The Etype Length and Etype Length Enable are used to enable parsing beyond the Ethertype defined by the ETQF entry, the Etype Length points to the following Ethertype in the packet to support extended Rx parsing.
- The *Rx Queue* field contains the absolute destination queue for the packet.
- The *1588 Time Stamp* field indicates that the packet should be time stamped according to the IEEE 1588 specification.
- The *Queue Enable* field enables forwarding Rx packets based on the Ether-type defined in this register. Refer to Section 7.1.2.1 on the impact and order of ETQF on Foxville queue selection algorithm.
- The Ethertype length field contains the size of the Ethertype in bytes.
- The Ethertype length Enable field enables the parsing of the Rx packets based on the Ethertype defined in this register.

Note: Software should not assign the same Ether-type value to different ETQF filters with different *Rx Queue* assignments.



Note: The Etype Length and Etype Length Enable should only be used when parsing beyond the defined Ethertype is required to enable Rx offloading for non L2 only packets.

Note: Queuing and Immediate interrupt decisions for an incoming packet that matches more than a single ETQF entry are done according to the setting of the last ETQF match.

7.1.2.4 2-Tuple Filters

These filters identify specific packets destined to a certain TCP/UDP port and implement a specific protocol. Each filter consists of a 2-tuple (protocol and destination TCP/UDP port) and forwards packets into one of the receive queues.

Foxville incorporates 8 such filters.

The 2-tuple filters are configured via the *TTQF* (See Section 8.10.3), *IMIR* (See Section 8.10.1) and *IMIR_EXT* (See Section 8.10.2) registers as follows (per filter):

- Protocol — Identifies the IP protocol, part of the 2-tuple queue filters. Enabled by a bit in the *TTQF.Mask* field.
- Destination port — Identifies the TCP/UDP destination port, part of the 2-tuple queue filters. Enabled by the *IMIR.PORT_BP* bit.
- Size threshold (*IMIREXT.Size_Thresh*) — Identifies the length of the packet that should trigger the filter. This is the length as received by the host, not including any part of the packet removed by hardware. Enabled by the *IMIREXT.Size_BP* field.
- Control Bits — Identify TCP flags that might be part of the filtering process. Enabled by the *IMIREXT.CtrlBit_BP* field.
- Rx queue — Determines the Rx queue for packets that match this filter:
  - The *TTQF.Rx Queue* field contains the queue serial number.
- Queue enable — Enables forwarding a packet that uses this filter to the queue defined in the *TTQF.Rx Queue* field.
- Mask — A 1-bit field that masks the L4 protocol check. The filter is a logical AND of the non-masked 2-tuple fields. If all 2-tuple fields are masked, the filter is not used for queue forwarding.
    - If more than one 2-tuple filter with the same priority matches the packet, the first filter (lowest ordinal number) is used in order to define the queue destination of this packet.
    - The immediate interrupt and 1588 actions are defined by the OR of all the matching filters.

7.1.2.5 Flex Filters

Foxville supports a total of 32 flexible filters. Each filter can be configured to recognize any arbitrary pattern within the first 128 bytes of the packet. To configure the flexible filters, software programs the mask values (required values and the minimum packet length), into the Flexible Host Filter Table (*FHFT* and *FHFT_EXT together with the select register - FHFTSL*, See Section 8.20.21, Section 8.20.22 and Section 8.20.23). These 32 flexible filters can be used as for wake-up or proxying when in D3 state or for queueing when in D0 state. Software must enable the filters in the *Wake Up Filter* Control (*WUFC and WUFC_EXT See* Section 8.20.2 and Section 8.20.3) register or Proxying Filter Control (*PROXYFC* see Section 8.20.9) for operation in D3 low power mode or in the *WUFC* register in D0 mode. In D0 mode these filters enable forwarding of packets that match up to 128 Bytes defined in the filter to one of the receive queues or as an indication to drop packets. In D3 mode these filters can be used for Wake-on-Lan as described in Section 5.5.3.1.8 or proxying as described in Section 5.8.

Once enabled, the flexible filters scan incoming packets for a match. If the filter encounters any byte in the packet where the mask bit is one and the byte doesn't match the value programmed in the Flexible Host Filter Table (*FHFT* or *FHFT_EXT*), then the filter fails that packet. If the filter reaches the required length without failing the packet, it forwards the packet to the appropriate receive queue. It ignores any mask bits set to one beyond the required length (defined in the Length field in the *FHFT* or *FHFT_EXT* registers).

Note: The flex filters are temporarily disabled when read from or written to by the host. Any packet received during a read or write operation is dropped. Filter operation resumes once the read or write access completes.

The flex filters are configured in D0 state via the *WUFC and WUFC_EXT*, *FHFT* and *FHFT_EXT* registers as follows (per filter):

- Byte Sequence to be compared - Program 128 Byte sequence, mask bits and *Length* field in FHFT and *FHFT_EXT* registers.
- Filter Priority - Program filter priority in queueing field in *FHFT* and *FHFT_EXT* registers.
- Receive queue - Program receive queue to forward packet in queueing field in *FHFT* and *FHFT_EXT* registers.
- Filter actions - Program immediate interrupt requirement in queueing field in *FHFT* and *FHFT_EXT* registers.
- Drop action - Packets that match the filter are dropped.
- Filter enable - Set *WUFC.FLEX_HQ* bit / *WUFC_EXT.FLEX_HQ* bit to 1 to enable flex filter operation in D0 state. Set appropriate *WUFC.FLX[n]* bit to 1 to enable specific flex filter.

Before entering D3 state software device driver programs the *FHFT* and *FHFT_EXT* filters for appropriate wake events and enables relevant filters by setting the *WUFC.FLX[n]* bit to 1 or the *PROXYFC.FLX[n]* bit to 1b. Following move to D0 state the software device driver programs the FHFT and FHFT_EXT filters for appropriate queueing decisions and enables the relevant filters by setting the *WUFC.FLX[n]* bit / *WUFC.FLX[n]* to 1b and the *WUFC.FLEX_HQ* bit to 1b.

Notes: If more than one flex filter with the same priority is matched by the packet, the first filter (lowest address) is used in order to define the queue destination of this packet.

The immediate interrupt action is defined by the OR of all the matching filters.

7.1.2.6 SYN Packet Filters

Foxville might forward TCP packets whose *SYN* flag is set into a separate queue. SYN packets are often used in SYN attacks to load the system with numerous requests for new connections. By filtering such packets to a separate queue, security software can monitor and act on SYN attacks.

SYN filters are configured via the SYNQF registers as follows:

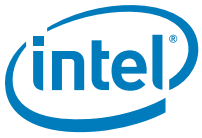
- Queue En — Enables forwarding of SYN packets to a specific queue.
- Rx Queue field — Contains the destination queue for the packet.

7.1.2.7 VLAN Priority Filters

Foxville can forward packets according to their VLAN priority to separate queues. Foxville supports the configuration of the destination queue per VLAN priority.

VLAN priority filters are configured via the VLANPQF registers as follows:

- Queue En — Enables forwarding of packets for each VLAN priority to a specific queue.



- Rx Queue field — Contains the destination queue for each VLAN priority packet.

7.1.2.8 MAC Address Filters

Foxville can forward packets according to their MAC address to separate queues. Foxville supports the configuration of the destination queue per MAC address.

MAC Address filters are configured via the RAL/H registers as follows:

- MAC address value - The MAC address value to be filtered
- Queue En — Enables forwarding of packets for each filtered MAC address to a specific queue.
- Rx Queue field — Contains the destination queue for each filtered MAC address.

7.1.2.9 Receive-Side Scaling (RSS)

RSS is a mechanism to distribute received packets into several descriptor queues. Software then assigns each queue to a different processor, sharing the load of packet processing among several processors.

Foxville uses RSS as one ingredient in its packet assignment policy (the others are the various filters for TSN). The RSS output is a RSS index. Foxville's global assignment uses these bits (or only some of the LSB bits) as part of the queue number.

RSS is enabled by the MRQC register. The RSS hash is reported only on the advanced receive descriptor and it multiplexed with UDP fragmentation parameters. Selection between these two status indications is done by the RXCSUM.PCSD bit setting.

When RSS is enabled, Foxville provides software with the following information as required by Microsoft* RSS specification or for device driver assistance:

- A Dword result of the Microsoft* RSS hash function, to be used by the stack for flow classification, is written into the receive packet descriptor (required by Microsoft* RSS).
- A 4-bit RSS *Type* field conveys the hash function used for the specific packet (required by Microsoft* RSS).

Figure 7-4 shows the process of computing an RSS output:

1. The receive packet is parsed into the header fields used by the hash operation (such as IP addresses, TCP port, etc.).
2. A hash calculation is performed. Foxville supports a single hash function, as defined by Microsoft* RSS. Foxville does not indicate to the software device driver which hash function is used. The 32-bit result is fed into the packet receive descriptor.
3. The seven LSB bits of the hash result are used as an index into a 128-entry indirection table. Each entry provides a 3-bit RSS output index.

When RSS is disabled, packets are assigned an RSS output index = zero. System software might enable or disable RSS at any time. While disabled, system software might update the contents of any of the RSS-related registers.

When multiple requests queues are enabled in RSS mode, un-decodable packets are assigned an RSS output index = zero. The 32-bit tag (normally a result of the hash function) equals zero.

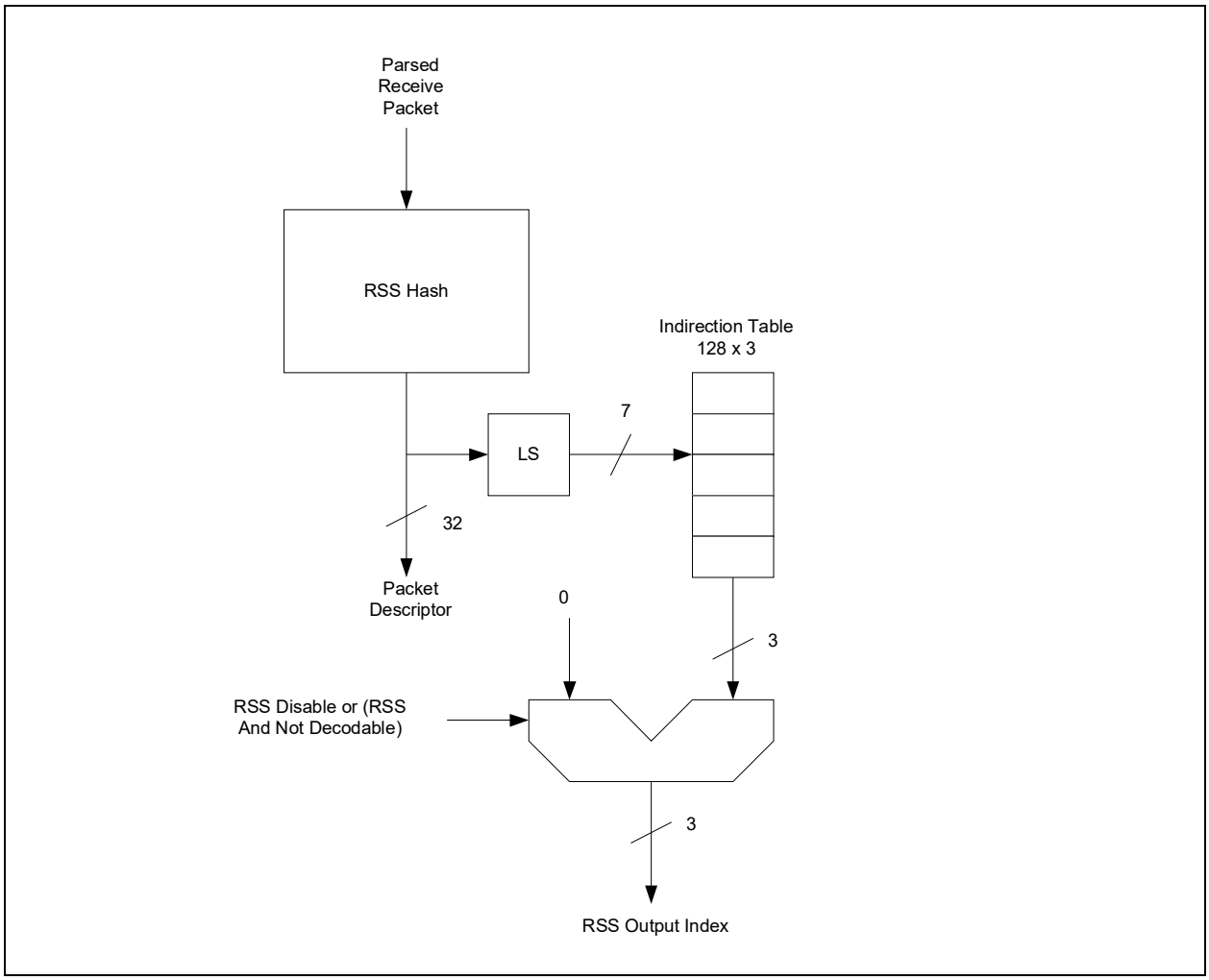


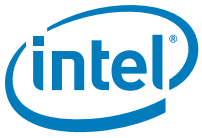
Figure 7-4. RSS Block Diagram

7.1.2.9.1 RSS Hash Function

Section 7.1.2.9.1 provides a verification suite used to validate that the hash function is computed according to Microsoft* nomenclature.

Foxville hash function follows Microsoft* definition. A single hash function is defined with several variations for the following cases:

- TcpIPv4 — Foxville parses the packet to identify an IPv4 packet containing a TCP segment per the criteria described later in this section. If the packet is not an IPv4 packet containing a TCP segment, RSS is not done for the packet.
- IPv4 — Foxville parses the packet to identify an IPv4 packet. If the packet is not an IPv4 packet, RSS is not done for the packet.



- TcpIPv6 — Foxville parses the packet to identify an IPv6 packet containing a TCP segment per the criteria described later in this section. If the packet is not an IPv6 packet containing a TCP segment, RSS is not done for the packet.
- TcpIPv6Ex — Foxville parses the packet to identify an IPv6 packet containing a TCP segment with extensions per the criteria described later in this section. If the packet is not an IPv6 packet containing a TCP segment, RSS is not done for the packet. Extension headers should be parsed for a *Home-Address-Option* field (for source address) or the *Routing-Header-Type-2* field (for destination address).
- IPv6Ex — Foxville parses the packet to identify an IPv6 packet. Extension headers should be parsed for a *Home-Address-Option* field (for source address) or the *Routing-Header-Type-2* field (for destination address). Note that the packet is not required to contain any of these extension headers to be hashed by this function. In this case, the IPv6 hash is used. If the packet is not an IPv6 packet, RSS is not done for the packet.
- IPv6 — Foxville parses the packet to identify an IPv6 packet. If the packet is not an IPv6 packet, receive-side-scaling is not done for the packet.

The following additional cases are not part of the Microsoft* RSS specification:

- UdpIPV4 — Foxville parses the packet to identify a packet with UDP over IPv4.
- UdpIPV6 — Foxville parses the packet to identify a packet with UDP over IPv6.
- UdpIPV6Ex — Foxville parses the packet to identify a packet with UDP over IPv6 with extensions.

A packet is identified as containing a TCP segment if all of the following conditions are met:

- The transport layer protocol is TCP (not UDP, ICMP, IGMP, etc.).
- The TCP segment can be parsed (such as IP options can be parsed, packet not encrypted).
- The packet is not fragmented (even if the fragment contains a complete TCP header).

Bits[31:16] of the Multiple Receive Queues Command (*MRQC)* register enable each of the above hash function variations (several can be set at a given time). If several functions are enabled at the same time, priority is defined as follows (skip functions that are not enabled):

IPv4 packet:

1. Try using the TcpIPv4 function.
2. Try using IPV4_UDP function.
3. Try using the IPv4 function.

IPv6 packet:

1. If TcpIPv6Ex is enabled, try using the TcpIPv6Ex function; else if TcpIPv6 is enabled try using the TcpIPv6 function.
2. If UdpIPv6Ex is enabled, try using UdpIPv6Ex function; else if UpdIPv6 is enabled try using UdpIPv6 function.
3. If IPv6Ex is enabled, try using the IPv6Ex function, else if IPv6 is enabled, try using the IPv6 function.

The following combinations are currently supported:

- Any combination of IPv4, TcpIPv4, and UdpIPv4.
- And/or.
- Any combination of either IPv6, TcpIPv6, and UdpIPv6 or IPv6Ex, TcpIPv6Ex, and UdpIPv6Ex.

When a packet cannot be parsed by the previously mentioned rules, it is assigned an RSS output index = zero. The 32-bit tag (normally a result of the hash function) equals zero.

The 32-bit result of the hash computation is written into the packet descriptor and also provides an index into the indirection table.

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 in the signature.
- A "^ "denotes bit-wise XOR operation of same-width vectors.
- @x-y denotes bytes x through y (including both of them) of the incoming packet, where byte 0 is the first byte of the IP header. In other words, it is considered that all byte-offsets as offsets into a packet where the framing layer header has been stripped out. Therefore, the source IPv4 address is referred to as @12-15, while the destination v4 address is referred to as @16-19.
- @x-y, @v-w denotes concatenation of bytes x-y, followed by bytes v-w, preserving the order in which they occurred in the packet.

All hash function variations (IPv4 and IPv6) follow the same general structure. Specific details for each variation are described in the following section. The hash uses a random secret key length of 320 bits (40 bytes); the key is typically supplied through the RSS Random Key Register (RSSRK).

The algorithm works 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, Intel's nomenclature assumes that the array is laid out as shown:

```
K[0] K[1] K[2] … K[k-1]
```

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 hash-input input[] of length N bytes (8N bits) and a random secret key K of 320 bits
Result = 0;
For each bit b in input[] {
if (b == 1) then Result ^= (left-most 32 bits of K);
shift K left 1 bit position;
}
return Result;
```

The following four pseudo-code examples are intended to help clarify exactly how the hash is to be performed in four cases, IPv4 with and without ability to parse the TCP header and IPv6 with an without a TCP header.

7.1.2.9.1.1 Hash for IPv4 with TCP

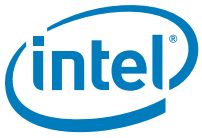
Concatenate SourceAddress, DestinationAddress, SourcePort, DestinationPort into one single byte-array, preserving the order in which they occurred in the packet:

```
Input[12] = @12-15, @16-19, @20-21, @22-23.
Result = ComputeHash(Input, 12);
```

7.1.2.9.1.2 Hash for IPv4 with UDP

Concatenate SourceAddress, DestinationAddress, SourcePort, DestinationPort into one single byte-array, preserving the order in which they occurred in the packet:

```
Input[12] = @12-15, @16-19, @20-21, @22-23.
Result = ComputeHash(Input, 12);
```



7.1.2.9.1.3 Hash for IPv4 without TCP

Concatenate SourceAddress and DestinationAddress into one single byte-array

```
Input[8] = @12-15, @16-19
Result = ComputeHash(Input, 8)
```

7.1.2.9.1.4 Hash for IPv6 with TCP

Similar to above:

```
Input[36] = @8-23, @24-39, @40-41, @42-43
Result = ComputeHash(Input, 36)
```

7.1.2.9.1.5 Hash for IPv6 with UDP

Similar to above:

```
Input[36] = @8-23, @24-39, @40-41, @42-43
Result = ComputeHash(Input, 36)
```

7.1.2.9.1.6 Hash for IPv6 without TCP

```
Input[32] = @8-23, @24-39
Result = ComputeHash(Input, 32)
```

7.1.2.9.2 Indirection Table

The *RETA* indirection table is a 128-entry structure, indexed by the seven LSB bits of the hash function output. Each entry of the table contains the following:

- Bits [2:0] - RSS index

System software might update the indirection table during run time. Such updates of the table are not synchronized with the arrival time of received packets. Therefore, it is not guaranteed that a table update takes effect on a specific packet boundary.

7.1.2.9.3 RSS Verification Suite

Assume that the random key byte-stream is:

```
0x6d, 0x5a, 0x56, 0xda, 0x25, 0x5b, 0x0e, 0xc2,
0x41, 0x67, 0x25, 0x3d, 0x43, 0xa3, 0x8f, 0xb0,
0xd0, 0xca, 0x2b, 0xcb, 0xae, 0x7b, 0x30, 0xb4,
0x77, 0xcb, 0x2d, 0xa3, 0x80, 0x30, 0xf2, 0x0c,
0x6a, 0x42, 0xb7, 0x3b, 0xbe, 0xac, 0x01, 0xfa
```

7.1.2.9.3.1 IPv4

Table 7-2. 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

Table 7-2. IPv4

Destination Address/Port	Source Address/Port	IPv4 Only	IPv4 With TCP
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

7.1.2.9.3.2 IPv6

The IPv6 address tuples are only for verification purposes and might not make sense as a tuple.

Table 7-3. 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	0xdde51bbf
fe80::200:f8ff:fe21:67cf (38024)	3ffe:1900:4545:3:200:f8ff:fe21:67cf (44251)	0x4b61e985	0x02d1feef

7.1.2.9.4 Association Through MAC Address

Software can program different values to the MAC filters (any bits in *RAH* or *RAL*) at any time. Foxville would respond to the change on a packet boundary but does not guarantee the change to take place at some precise time.

7.1.3 Receive Data Storage

7.1.3.1 Host Buffers

Each descriptor points to a one or more memory buffers that are designated by the software device driver to store packet data.

The size of the buffer can be set using either the generic *RCTL.BSIZE* field, or the per queue *SRRCTL[n].BSIZEPACKET* field.

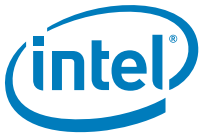
If *SRRCTL[n].BSIZEPACKET* is set to zero for any queue, the buffer size defined by *RCTL.BSIZE* is used. Otherwise, the buffer size defined by *SRRCTL[n].BSIZEPACKET* is used.

If the receive buffer size is selected by bit settings in the Receive Control (*RCTL.BSIZE*) buffer sizes of 256, 512, 1024, and 2048 bytes are supported.

If the receive buffer size is selected by *SRRCTL[n].BSIZEPACKET,* buffer sizes of 1KB to 127 KB are supported with a resolution of 1 KB.

In addition, for advanced descriptor usage the *SRRCTL.BSIZEHEADER* field is used to define the size of the buffers allocated to headers. Header Buffer sizes of 64 bytes to2048 bytes with a resolution of 64 bytes are supported.

Foxville places no alignment restrictions on receive memory buffer addresses. This is desirable in situations where the receive buffer was allocated by higher layers in the networking software stack, as these higher layers might have no knowledge of a specific device's buffer alignment requirements.



Note: When the *No-Snoop Enable* bit is used in advanced descriptors, the buffer address is 16-bit (2-byte) aligned.

7.1.3.2 On-Chip Receive Buffer

Foxville allocates by default a 36 KB on-chip packet buffer. The buffer is used to store packets until they are forwarded to the host. The receive packet buffer is allocated to high priority traffic and low priority traffic. Packets from the high priority packet buffer are always posted to host memory before packets from the low priority buffer. The sizes of the high priority and low priority buffers are controlled by the RXPBSIZE_EXP and RXPBSIZE_BE fields in the RXPBSIZE register. The sum of the low and high priority packet buffers must not be set to more than 36 KB (which is the total size of the receive packet buffer).

When preemption is enabled by the PREEMPT_ENA flag in TQAVCTRL register, Express traffic is routed to the high priority packet buffer and Best Effort traffic is routed to the low priority packet buffer.

When preemption is not enabled, the received packets are directed to these packet buffers by the VLAN priority in the packets. Packets with no VLAN header are handled the same as VLAN priority zero. VLAN priorities are mapped to one of the two packet buffers by the VP**xx**PBSEL fields in the VLANPQF register, while '**xx**' equals to 0...7 for VLAN priority 0...7 respectively.

7.1.3.3 On-chip Descriptor Buffers

Foxville contains a 16 descriptor cache for each receive queue used to reduce the latency of packet processing and to optimize the usage of PCIe bandwidth by fetching and writing back descriptors in bursts. The fetch and write-back algorithm are described in Section 7.1.4.3 and Section 7.1.4.4.

7.1.4 Receive Descriptors

7.1.4.1 Legacy Receive Descriptor Format

A receive descriptor is a data structure that contains the receive data buffer address and fields for hardware to store packet information. If *SRRCTL[n].DESCTYPE* = 000b, Foxville uses the legacy Receive descriptor described in this section.

7.1.4.1.1 Legacy Receive Descriptors - Read Format

Table 7-4 shows the receive descriptor. This is the format that software writes to the descriptor queue and hardware reads from the descriptor queue in host memory. Hardware writes back the descriptor in a different format, shown in Table 7-5.

Table 7-4. Legacy Receive Descriptor (RDESC) Layout

	63	0
0	Buffer Address [63:0]	
8	Reserved Zero	

Packet Buffer Address (64) - Physical address of the packet buffer.

7.1.4.1.2 Legacy Receive Descriptors - Write-Back Format

Table 7-5. Legacy Receive Descriptor (RDESC) Layout

	63 48 47 40 39 32 31 16 15 0				
0	Reserved (Buffer Address)				
8	VLAN Tag	Errors	Status	Fragment Checksum	Length

After receiving a packet for Foxville, hardware stores the packet data into the indicated buffer and writes the length, packet checksum, status, errors, and status fields.

Length Field (16)

Length covers the data written to a receive buffer including CRC bytes (if any). Software must read multiple descriptors to determine the complete length for a packet that spans multiple receive buffers.

Fragment Checksum (16)

This field is used to provide the fragment checksum value. This field equals to the unadjusted 16-bit ones complement of the packet. Checksum calculation starts at the L4 layer (after the IP header) until the end of the packet excluding the CRC bytes. In order to use the fragment checksum assist to offload L4 checksum verification, software might need to back out some of the bytes in the packet. For more details see Section 7.1.7.2

Status Field (8)

Status information indicates whether the descriptor has been used and whether the referenced buffer is the last one for the packet. See Table 7-6 for the layout of the *Status* field. Error status information is shown in Table 7-10.

Table 7-6. Receive Status (RDESC.STATUS) Layout

7	6	5	4	3	2	1	0
PIF	IPCS	L4CS	UDPCS	VP	Rsv	EOP	DD

- PIF (bit 7) - Passed imperfect filter only
- IPCS (bit 6) - IPv4 checksum calculated on packet
- L4CS (bit 5) - L4 (UDP or TCP) checksum calculated on packet
- UDPCS (bit 4) - UDP checksum or IP payload checksum calculated on packet.
- VP (bit 3) - Packet is 802.1Q (matched VET); indicates strip VLAN in 802.1Q packet
- RSV (bit 2) - Reserved
- EOP (bit 1) - End of packet
- DD (bit 0) - Descriptor done

EOP and DD

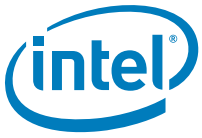


Table 7-7 lists the meaning of these bits:

Table 7-7. Receive Status Bits

DD	EOP	Description
0b	0b	Software setting of the descriptor when it hands it off to the hardware.
0b	1b	Reserved (invalid option).
1b	0b	A completion status indication for a non-last descriptor of a packet that spans across multiple descriptors. In a single packet case, DD indicates that the hardware is done with the descriptor and its buffers. Only the *Length* fields are valid on this descriptor.
1b	1b	A completion status indication of the entire packet. Note that software Might take ownership of its descriptors. All fields in the descriptor are valid (reported by the hardware).

VP Field

The *VP* field indicates whether the incoming packet's type matches the VLAN Ethernet Type programmed in the *VET* Register. For example, if the packet is a VLAN (802.1Q) type, it is set if the packet type matches *VET* and *CTRL.VME* is set (VLAN mode enabled). It also indicates that VLAN has been stripped from the 802.1Q packet. For more details, see Section 7.4.

IPCS (IPv4 Checksum), L4CS (L4 Checksum), and UDPCS (UDP Checksum)

The meaning of these bits is listed in Table 7-8:

Table 7-8. IPCS, L4CS, and UDPCS

L4CS	UDPCS	IPCS	Functionality
0b	0b	0b	Hardware does not provide checksum offload. Special case: Hardware does not provide UDP checksum offload for IPV4 packet with UDP checksum = 0b
1b	0b	1b / 0b	Hardware provides IPv4 checksum offload if IPCS is active and TCP checksum is offload. A pass/fail indication is provided in the *Error* field – IPE and L4E. See the PKTTYPE table for supported packet types.
0b	1b	1b / 0b	Hardware provides IPv4 checksum offload if IPCS is active and UDP checksum is offload. A pass/fail indication is provided in the *Error* field – IPE and L4E. See PKTTYPE table for supported packet types.

Refer to Table 7-20 for a description of supported packet types for receive checksum offloading. Unsupported packet types do not have the *IPCS* or *L4CS* bits set. IPv6 packets do not have the *IPCS* bit set, but might have the *L4CS* bit set if Foxville recognized the TCP or UDP packet.

PIF

Hardware supplies the *PIF* field to expedite software processing of packets. Software must examine any packet with *PIF* bit set to determine whether to accept the packet. If the *PIF* bit is clear, then the packet is known to be destined to this station, so software does not need to look at the packet contents. Multicast / Unicast packets passing only the MTA set the *PIF* bit. In addition, the following condition causes *PIF* to be cleared:

- The DA of the packet is a multicast address and promiscuous multicast is set (*RCTL.MPE* = 1b).
- The DA of the packet is a broadcast address and accept broadcast mode is set (*RCTL.BAM* = 1b)

A MAC control frame forwarded to the host (*RCTL.PMCF* = 0b) that does not match any of the exact filters, has the *PIF* bit set.

Error Field (8)

Most error information appears only when the *store-bad-packet* bit (*RCTL.SBP*) is set and a bad packet is received. See Table 7-9 for a definition of the possible errors and their bit positions.

Table 7-9. RXE, IPE and L4E

7	6	5	4	3	2	1	0
RXE	IPE	L4E	Reserved				

- RXE (bit 7) - RX Data Error
- IPE (bit 6) - IPv4 Checksum Error
- L4E (bit 5) - TCP/UDP Checksum Error
- Reserved (bit 4:0)

IPE/L4E

The IP and TCP/UDP checksum error bits from Table 7-9 are valid only when the IPv4 or TCP/UDP checksum(s) is performed on the received packet as indicated via IPCS and L4CS. These, along with the other error bits, are valid only when the *EOP* and *DD* bits are set in the descriptor.

Note: Receive checksum errors have no effect on packet filtering.

If receive checksum offloading is disabled (*RXCSUM.IPOFLD* and *RXCSUM.TUOFLD*), the *IPE* and *L4E* bits are 0b.

RXE

The RXE error bit is asserted in the following case:

1. CRC error is detected. CRC can be a result of reception of /V/ symbol on the TBI interface (see section 3.6.2.3.3) or assertion of RxERR on the MII/GMII interface or bad EOP or lose of sync during packet reception. Packets with a CRC error are posted to host memory only when *store-bad-packet* bit (*RCTL.SBP*) is set.

VLAN Tag Field (16)

Hardware stores additional information in the receive descriptor for 802.1Q packets. If the packet type is 802.1Q (determined when a packet matches *VET* and *CTRL.VME* = 1b), then the *VLAN Tag* field records the VLAN information and the four-byte VLAN information is stripped from the packet data storage. Otherwise, the *VLAN Tag* field contains 0x0000. The rule for *VLAN tag* is to use network ordering (also called big endian). It appears in the following manner in the descriptor:

Table 7-10. VLAN Tag Field Layout (for 802.1Q Packet)

15 13	12	11 0
PRI	CFI	VLAN

7.1.4.2 Advanced Receive Descriptors

7.1.4.2.1 Advanced Receive Descriptors (RDESC) - Read Format

Table 7-11 shows the receive descriptor. This is the format that software writes to the descriptor queue and hardware reads from the descriptor queue in host memory. Hardware writes back the descriptor in a different format, shown in Table 7-12.

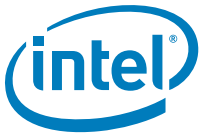


Table 7-11. RDESC Descriptor Read Format

	63 1	0
0	Packet Buffer Address [63:1]	A0/NSE
8	Header Buffer Address [63:1]	DD

Packet Buffer Address (64) - Physical address of the packet buffer. The lowest bit is either A0 (LSB of address) or NSE (No-Snoop Enable), depending on bit *RXCTL.RXdataWriteNSEn* of the relevant queue. See Section 3.1.5.5.4.

Header Buffer Address (64) - Physical address of the header buffer. The lowest bit is *DD*.

Note: Foxville does not support null descriptors (a descriptor with a packet or header address that is always equal to zero).

When software sets the *NSE* bit in the receive descriptor, Foxville places the received packet associated with this descriptor in memory at the packet buffer address with *NSE* set in the PCIe attribute fields. *NSE* does not affect the data written to the header buffer address.

When a packet spans more than one descriptor, the header buffer address is not used for the second, third, etc. descriptors; only the packet buffer address is used in this case.

NSE is enabled for packet buffers that the software device driver knows have not been touched by the processor since the last time they were used, so the data cannot be in the processor cache and snoop is always a miss. Avoiding these snoop misses improves system performance. No-snoop is particularly useful when the DMA engine is moving the data from the packet buffer into application buffers, and the software device driver is using the information in the header buffer for its work with the packet.

Note: When No-Snoop Enable is used, relaxed ordering should also be enabled with *CTRL_EXT.RO_DIS*.

7.1.4.2.2 Advanced Receive Descriptors (RDESC) - Write-back Format

When Foxville writes back the descriptors, it uses the descriptor format shown in Table 7-12.

Note: *SRRCTL[n]. DESCTYPE* must be set to a value other than 000b for Foxville to write back the special descriptors.

Table 7-12. RDESC Descriptor Write-back Format

	63 48	47 35	34 32	31	30 21	20	19	18 17	16 4	3 0
0	RSS Hash Value/ {Fragment Checksum, IP identification}			SPH	HDR_LEN[9:0]	HDR_LEN[11:10]		RSV	Packet Type	RSS Type
8	VLAN Tag	PKT_LEN		Extended Error		Extended Status				

RSS Type (4)

Table 7-13. RSS Type

Packet Type	Description
0x0	No hash computation done for this packet.
0x1	HASH_TCP_IPV4
0x2	HASH_IPV4
0x3	HASH_TCP_IPV6
0x4	HASH_IPV6_EX
0x5	HASH_IPV6
0x6	HASH_TCP_IPV6_EX
0x7	HASH_UDP_IPV4
0x8	HASH_UDP_IPV6
0x9	HASH_UDP_IPV6_EX
0xA:0xF	Reserved

Foxville must identify the packet type and then choose the appropriate RSS hash function to be used on the packet. The RSS type reports the packet type that was used for the RSS hash function.

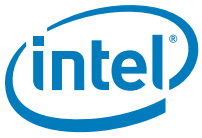
Packet Type (13)

- VPKT (bit 12) - VLAN Packet indication
- L2 Packet (bit 11) - L2 packet indication (not L3 or L4 packet), if this bit is set along with a higher layer indication it indicates the ETQF type is valid. The matched ETQF filter is indicated in bits 7:0 of this field.
- When the L2 Packet flag (bit 11) is cleared, the 11 LSB bits of the packet type reports the packet type identified by the hardware as follows:

Table 7-14. Packet Type LSB Bits (11:10)

Bit Index	Bit 11 = 0b
0	IPV4 - Indicates IPv4 header present[1]
1	IPV4E - Indicates IPv4 Header includes IP options[1]
2	IPV6 - Indicates IPv6 header present[1 2 3]
3	IPV6E - Indicates IPv6 Header includes extensions[1 2 3]
4	TCP - Indicates TCP header present[1 3 4]
5	UDP - Indicates UDP header present[1 3 4]
6	SCTP - Indicates SCTP header present[1 3 4]
7	NFS - Indicates NFS header present[1 3 4]
10:8	EtherType - ETQF register index that matches the packet. Special types might be defined for 1588, 802.1X, LLDP BCN or any other requested type.Ethertype - ETQF register index that matches the packet. Special types might be defined for 1588, 802.1x, 1722, LLDP or other requested EtherTypes

1. On unsupported tunneled frames only packet types of external IP header are set if detected.
2. When a packet is fragmented then the internal packet type bits on a supported tunneled packet (IPv6 tunneled in IPv4 only) won't be set.



3. On supported tunneled frames (IPv6 tunneled in IPv4 only) then all the internal Packet types are set if detected (IPV6, IPV6E, TCP, UDP, SCTP and NFS)
4. When a packet is fragmented the TCP, UDP, SCTP and NFS bits won't be set.

RSV: Reserved.

HDR_LEN (10) - The length (bytes) of the header as parsed by Foxville. In split mode when HBO (Header Buffer Overflow) is set in the Extended error field, the *HDR_LEN* can be greater then zero though nothing is written to the header buffer. In header replication mode, the *HDR_LEN* field does not reflect the size of the data actually stored in the header buffer because Foxville fills the buffer up to the size configured by *SRRCTL[n].BSIZEHEADER*, which might be larger than the header size reported here. This field is only valid in the first descriptor of a packet and should be ignored in all subsequent descriptors.

Note: When the packet is time stamped and the time stamp is placed at the beginning of the buffer the *RDESC.HDR_LEN* field is updated with the additional time stamp bytes (16 bytes). For further information see Section 7.1.7.

Packet types supported by the header split and header replication are listed in Appendix N.1. Other packet types are posted sequentially in the host packet buffer. Each line in Table 7-15 has an enable bit in the PSRTYPE register. When one of the bits is set, the corresponding packet type is split. If the bit is not set, a packet matching the header layout is not split.

Header split and replication is described in Section 7.1.5 while the packet types for this functionality are enabled by the *PSRTYPE[n]* registers (Section 8.9.2).

Note: The header of a fragmented IPv6 packet is defined before the fragmented extension header.

SPH (1) - Split Header - When set, indicates that the *HDR_LEN* field reflects the length of the header found by hardware. If cleared, the *HDR_LEN* field should be ignored, unless *SRRCTL[n].DESCTYPE* is set to *Split – always use header buffer* mode and *PKT_LEN* = 0. In this case, the *HDR_LEN* reflects the size of the packet, even if *SPH* bit is cleared.

In the case where *SRRCTL[n].DESCTYPE* is set to *Header replication mode*, *SPH* bit is set but the *HDR_LEN* field does not reflect the size of the data actually stored in the header buffer, because Foxville fills the buffer up to the size configured by *SRRCTL[n].BSIZEHEADER*.

RSS Hash / {Fragment Checksum, IP identification} (32)

This field has multiplexed functionality according to the received packet type (reported on the *Packet Type* field in this descriptor) and device setting.

Fragment Checksum (16-Bit; 63:48)

The fragment checksum word contains the unadjusted one's complement checksum of the IP payload and is used to offload checksum verification for fragmented UDP packets as described in Section 7.1.7.2. This field is mutually exclusive with the RSS hash. It is enabled when the *RXCSUM.PCSD* bit is cleared and the *RXCSUM.IPPCSE* bit is set.

IP identification (16-Bit; 47:32)

The IP identification word identifies the IP packet to whom this fragment belongs and is used to offload checksum verification for fragmented UDP packets as described in Section 7.1.7.2. This field is mutually exclusive with the RSS hash. It is enabled when the *RXCSUM.PCSD* bit is cleared and the *RXCSUM.IPPCSE* bit is set.

RSS Hash Value (32)

The RSS hash value is required for RSS functionality as described in Section 7.1.2.7. This bit is mutually exclusive with the fragment checksum. It is enabled when the *RXCSUM.PCSD* bit is set.

Extended Status (20)

Status information indicates whether the descriptor has been used and whether the referenced buffer is the last one for the packet. Table 7-15 lists the extended status word in the last descriptor of a packet (*EOP* is set). Table 7-16 lists the extended status word in any descriptor but the last one of a packet (*EOP* is cleared).

Table 7-15. Receive Status (RDESC.STATUS) Layout of the Last Descriptor

19	18 16	15	14 13	12	11	10
MC	Reserved	TSIP	SMD Type	Strip CRC	LLINT	UDPV

VEXT	Rsv	PIF	IPCS	L4I	UDPCS	VP	Rsv	EOP	DD
9	8	7	6	5	4	3	2	1	0

Table 7-16. Receive Status (RDESC.STATUS) Layout of Non-Last Descriptor

19 2	1	0
Reserved	EOP = 0b	DD

MC (19) - Packet received from MC. The MC bit is set to indicate the packet was sent by the local MC. Bit is cleared if packet arrives from the network. For more details see Section 10.3.3.

TSIP (15) - Timestamp in packet. The *Timestamp In Packet* bit is set to indicate that the received packet arrival time was captured by the hardware and the timestamp was placed in the receive buffer. For further details see Section 7.1.7. The TSIP is always cleared for packets that were preempted.

SMD Type (14:13) - Indicates the type of SMD in the received packet as follow:

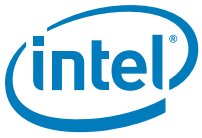
- 00b - SMD equals to the standard SFD
- 01b - SMD indicates a "Verify" packet
- 10b - SMD indicates a "Response" packet
- 11b - Complete Preempted packet

PIF (7), IPCS(6), UDPCS(4), VP(3), EOP (1), DD (0) - These bits are described in the legacy descriptor format in Section 7.1.4.

L4I (5) - This bit indicates that an L4 integrity check was done on the packet, either TCP checksum, UDP checksum or SCTP CRC checksum. This bit is valid only for the last descriptor of the packet. An error in the integrity check is indicated by the *L4E* bit in the error field. The type of check done can be induced from the packet type bits 4, 5 and 6. If bit 4 is set, a TCP checksum was done. If bit 5 is set a UDP checksum was done, and if bit 6 is set, a SCTP CRC checksum was done.

VEXT (9) - First VLAN is found on a double VLAN packet. This bit is valid only when *CTRL_EXT.EXT_VLAN* is set. For more details see Section 7.4.5.

UDPV (10) - This bit indicates that the incoming packet contains a valid (non-zero value) checksum field in an incoming first fragment UDP IPv4 packet. This means that the Fragment Checksum field in the receive descriptor contains the IP payload checksum as described in Section 7.1.7.2. When this field is cleared in the first fragment that contains the UDP header,



means that the packet does not contain a valid UDP checksum and the fragment checksum field in the Rx descriptor should be ignored. This field is always cleared in incoming fragments that do not contain the UDP header or in non fragmented packet.

LLINT (11) - This bit indicates that the packet caused an immediate interrupt via the low latency interrupt mechanism.

Strip CRC (12) - This bit indicates that Ethernet CRC has been stripped from incoming packet. Strip CRC operation is defined by the *RCTL.SECRC* bit.

Reserved (2, 8) - Reserved at zero.

Extended Error (12)

Table 7-17 and the text that follows describes the possible errors reported by hardware.

Table 7-17. Receive Errors (RDESC.ERRORS) Layout

11	10	9	8 7	6 4	3	2 0
RXE	IPE	L4E		Reserved	HBO	Reserved

RXE (bit 11)

RXE is described in the legacy descriptor format in Section 7.1.4.

IPE (bit 10)

The IPE error indication is described in the legacy descriptor format in Section 7.1.4.

L4E (bit 9)

L4 error indication - When set, indicates that hardware attempted to do an L4 integrity check as described in the *L4I* bit, but the check failed.

Reserved (bits 8:7)

Reserved (bits 6:4)

HBO (bit 3) - Header Buffer Overflow

Note: The *HBO* bit is relevant only if *SPH* is set.

1. In both header replication modes, *HBO* is set if the header size (as calculated by hardware) is bigger than the allocated buffer size (*SRRCTL.BSIZEHEADER*) but the replication still takes place up to the header buffer size. Hardware sets this bit in order to indicate to software that it needs to allocate bigger buffers for the headers.
2. In header split mode, when *SRRCTL[n] BSIZEHEADER* is smaller than *HDR_LEN*, then *HBO* is set to 1b, In this case, the header is not split. Instead, the header resides within the host packet buffer. The *HDR_LEN* field is still valid and equal to the calculated size of the header. However, the header is not copied into the header buffer.
3. In header split mode, always use header buffer mode, when *SRRCTL[n] BSIZEHEADER* is smaller than *HDR_LEN*, then *HBO* is set to 1b. In this case, the header buffer is used as part of the data buffers and contains the first BSIZEHEADER bytes of the packet. The *HDR_LEN* field is still valid and equal to the calculated size of the header.

Note: Most error information appears only when the *store–bad–packet* bit (*RCTL.SBP*) is set and a bad packet is received.

Reserved (bits 2:0) - Reserved

PKT_LEN (16)

Number of bytes existing in the host packet buffer

The length covers the data written to a receive buffer including CRC bytes (if any). Software must read multiple descriptors to determine the complete length for packets that span multiple receive buffers. If *SRRCTL.DESC_TYPE* = 4 (advanced descriptor header replication large packet only) and the total packet length is smaller than the size of the header buffer (no replication is done), this field continues to reflect the size of the packet, although no data is written to the packet buffer. Otherwise, if the buffer is not split because the header is bigger than the allocated header buffer, this field reflects the size of the data written to the first packet buffer (header and data).

Note: When the packet is time stamped and the time stamp is placed at the beginning of the buffer, the *RDESC.PKT_LEN* field is updated with the additional time stamp bytes (16 bytes). For further information see Section 7.1.7.

VLAN Tag (16)

These bits are described in the legacy descriptor format in Section 7.1.4.

7.1.4.3 Receive Descriptor Fetching

The fetching algorithm attempts to make the best use of PCIe bandwidth by fetching a cache-line (or more) descriptor with each burst. The following paragraphs briefly describe the descriptor fetch algorithm and the software control provided.

When the *RXDCTL[n].ENABLE* bit is set and the on-chip descriptor cache is empty, a fetch happens as soon as any descriptors are made available (Host increments the *RDT[n]* tail pointer). When the on-chip buffer is nearly empty (defined by *RXDCTL.PTHRESH*), a prefetch is performed each time enough valid descriptors (defined by *RXDCTL.HTHRESH*) are available in host memory.

When the number of descriptors in host memory is greater than the available on-chip descriptor cache, Foxville might elect to perform a fetch that is not a multiple of cache-line size. Hardware performs this non-aligned fetch if doing so results in the next descriptor fetch being aligned on a cache-line boundary. This enables the descriptor fetch mechanism to be more efficient in the cases where it has fallen behind software.

All fetch decisions are based on the number of descriptors available and do not take into account any split of the transaction due to bus access limitations.

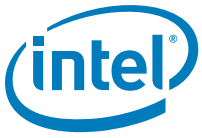
Note: Foxville NEVER fetches descriptors beyond the descriptor tail pointer.

7.1.4.4 Receive Descriptor Write-back

Processors have cache-line sizes that are larger than the receive descriptor size (16 bytes). Consequently, writing back descriptor information for each received packet would cause expensive partial cache-line updates. A receive descriptor packing mechanism minimizes the occurrence of partial line write-backs.

To maximize memory efficiency, receive descriptors are packed together and written as a cache-line whenever possible. Descriptors write-backs accumulate and are opportunistically written out in cache line-oriented chunks, under the following scenarios:

- *RXDCTL[n].WTHRESH* descriptors have been used (the specified maximum threshold of unwritten used descriptors has been reached).
- The receive timer expires (*EITR*) - in this case all descriptors are flushed ignoring any cache-line boundaries.



- Explicit software flush (*RXDCTL.SWFLS*).
- Dynamic packets - if at least one of the descriptors that are waiting for write-back are classified as packets requiring immediate notification the entire queue is flushed out.

When the number of descriptors specified by *RXDCTL[n].WTHRESH* have been used, they are written back regardless of cache-line alignment. It is therefore recommended that *RXDCTL[n].WTHRESH* be a multiple of cache-line size. When the receive timer (*EITR*) expires, all used descriptors are forced to be written back prior to initiating the interrupt, for consistency. Software might explicitly flush accumulated descriptors by writing the *RXDCTL[n]* register with the *SWFLS* bit set.

When Foxville does a partial cache-line write-back, it attempts to recover to cache-line alignment on the next write-back.

For applications where the latency of received packets is more important than the bus efficiency and the CPU utilization, an *EITR* value of zero can be used. In this case, each receive descriptor are written to the host immediately. If *RXDCTL[n].WTHRESH* equals zero, then each descriptor are written back separately;, otherwise, write back of descriptors can be coalesced if descriptor accumulates in the internal descriptor ring due to bandwidth constrains.

All write-back decisions are based on the number of descriptors available and do not take into account any split of the transaction due to bus access limitations.

7.1.4.5 Receive Descriptor Ring Structure

Figure 7-5 shows the structure of each of the 4 receive descriptor rings. Hardware maintains 4 circular queues of descriptors and writes back used descriptors just prior to advancing the head pointer(s). Head and tail pointers wrap back to base when size descriptors have been processed.

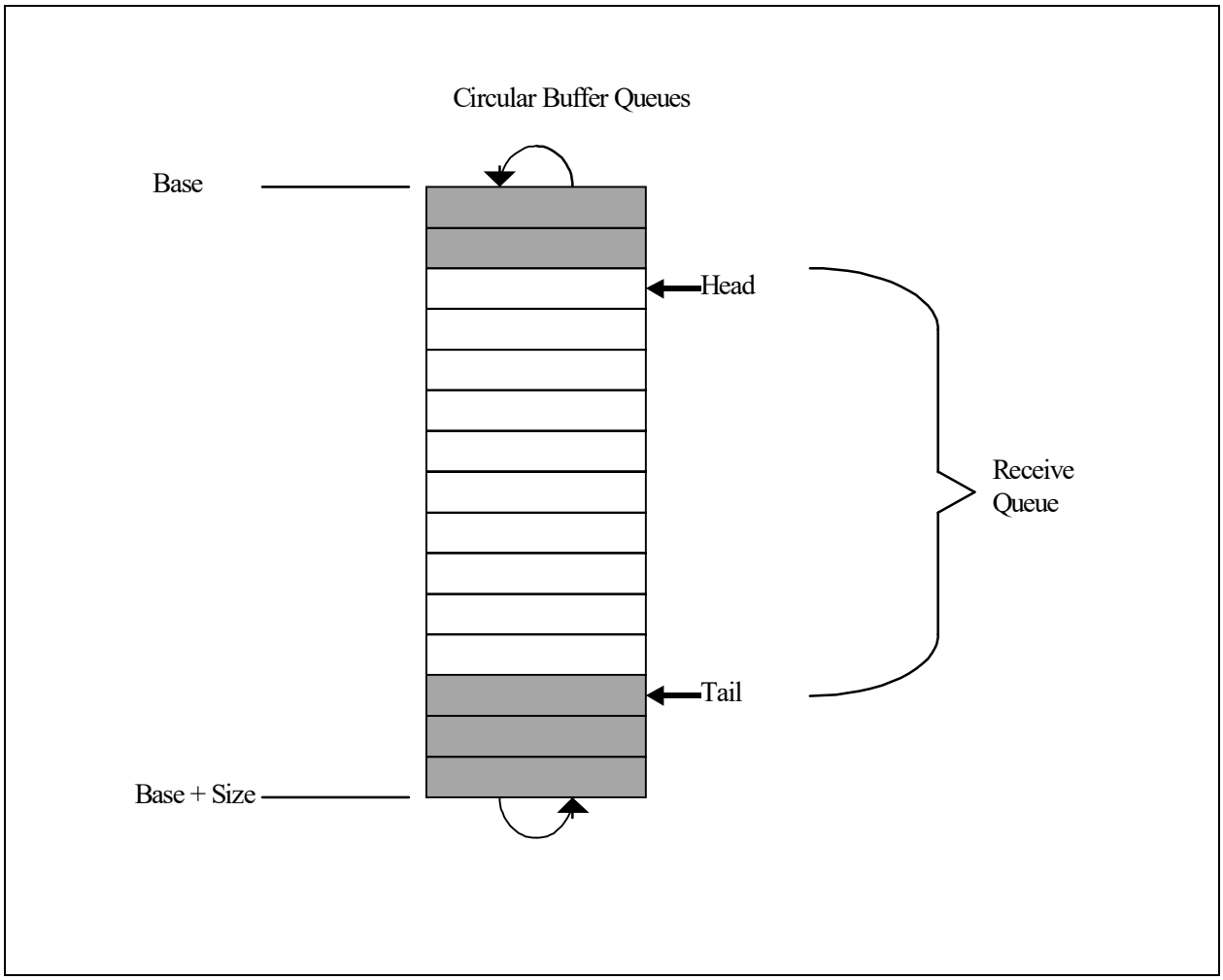
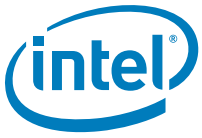


Figure 7-5. Receive Descriptor Ring Structure

Software inserts receive descriptors by advancing the tail pointer(s) to refer to the address of the entry just beyond the last valid descriptor. This is accomplished by writing the descriptor tail register(s) with the offset of the entry beyond the last valid descriptor. The hardware adjusts its internal tail pointer(s) accordingly. As packets arrive, they are stored in memory and the head pointer(s) is incremented by hardware. When the head pointer(s) is equal to the tail pointer(s), the queue(s) is empty. Hardware stops storing packets in system memory until software advances the tail pointer(s), making more receive buffers available.

The receive descriptor head and tail pointers reference to 16-byte blocks of memory. Shaded boxes in Figure 7-5 represent descriptors that have stored incoming packets but have not yet been recognized by software. Software can determine if a receive buffer is valid by reading the descriptors in memory. Any descriptor with a non-zero DD value has been processed by the hardware and is ready to be handled by the software.



Note: The head pointer points to the next descriptor that is written back. After the descriptor write-back operation completes, this pointer is incremented by the number of descriptors written back. Hardware owns all descriptors between [head... tail]. Any descriptor not in this range is owned by software.

The receive descriptor rings are described by the following registers:

- Receive Descriptor Base Address (*RDBA3* to *RDBA0*) register:
  This register indicates the start of the descriptor ring buffer. This 64-bit address is aligned on a 16-byte boundary and is stored in two consecutive 32-bit registers. Note that hardware ignores the lower 4 bits.
- Receive Descriptor Length (*RDLEN3* to *RDLEN0*) registers:
  This register determines the number of bytes allocated to the circular buffer. This value must be a multiple of 128 (the maximum cache-line size). Since each descriptor is 16 bytes in length, the total number of receive descriptors is always a multiple of eight.
- Receive Descriptor Head (*RDH3* to *RDH0*) registers:
  This register holds a value that is an offset from the base and indicates the in-progress descriptor. There can be up to 64 KB, 8 KB descriptors in the circular buffer. Hardware maintains a shadow copy that includes those descriptors completed but not yet stored in memory.
- Receive Descriptor Tail (*RDT3* to *RDT0*) registers:
  This register holds a value that is an offset from the base and identifies the location beyond the last descriptor hardware can process. This is the location where software writes the first new descriptor.

If software statically allocates buffers, uses legacy receive descriptors, and uses memory read to check for completed descriptors, it has to zero the status byte in the descriptor before bumping the tail pointer to make it ready for reuse by hardware. Zeroing the status byte is not a hardware requirement but is necessary for performing an in-memory scan.

All the registers controlling the descriptor rings behavior should be set before receive is enabled, apart from the tail registers that are used during the regular flow of data.

7.1.4.5.1 Low Receive Descriptors Threshold

As described above, the size of the receive queues is measured by the number of receive descriptors. During run time the software processes completed descriptors and then increments the Receive Descriptor Tail registers (*RDT*). At the same time, hardware might post new packets received from the LAN incrementing the Receive Descriptor Head registers (*RDH*) for each used descriptor.

The number of usable (free) descriptors for the hardware is the distance between Tail and Head registers. When the Tail reaches the Head, there are no free descriptors and further packets might be either dropped or block the receive FIFO. In order to avoid this behavior, Foxville might generate a low latency interrupt (associated with the relevant receive queue) once the amount of free descriptors is less or equal than the threshold. The threshold is defined in 16 descriptors granularity per queue in the *SRRCTL[n].RDMTS* field.

7.1.5 Header Splitting and Replication

7.1.5.1 Purpose

This feature consists of splitting or replicating packet's header to a different memory space. This helps the host to fetch headers only for processing: headers are replicated through a regular snoop transaction in order to be processed by the host CPU. It is recommended to perform this transaction with a software-prefetch.

The packet (header and payload) is stored in memory through a (optionally) non-snoop transaction. Later, a Crystal Beach (North Bridge or IOH DMA) transaction moves the payload from the software device driver buffer to application memory or it is moved using a normal memory copy operation.

Foxville supports header splitting in several modes:

- Legacy mode: legacy descriptors are used; headers and payloads are not split.
- Advanced mode, no split: advanced descriptors are in use; header and payload are not split.
- Advanced mode, split: advanced descriptors are in use; header and payload are split to different buffers. If the packet cannot be split, only the packet buffer is used.
- Advanced mode, replication: advanced descriptors are in use; header is replicated in a separate buffer and also in a payload buffer.
- Advanced mode, replication, conditioned by packet size: advanced descriptors are in use; replication is performed only if the packet is larger than the header buffer size.
- Advanced mode, split, always use header buffer: advanced descriptors are in use; header and payload are split to different buffers. If no split is done, the first part of the packet is stored in the header buffer.

7.1.5.2 Description

In Figure 7-6 and Figure 7-7, the header splitting and header replication modes are shown.

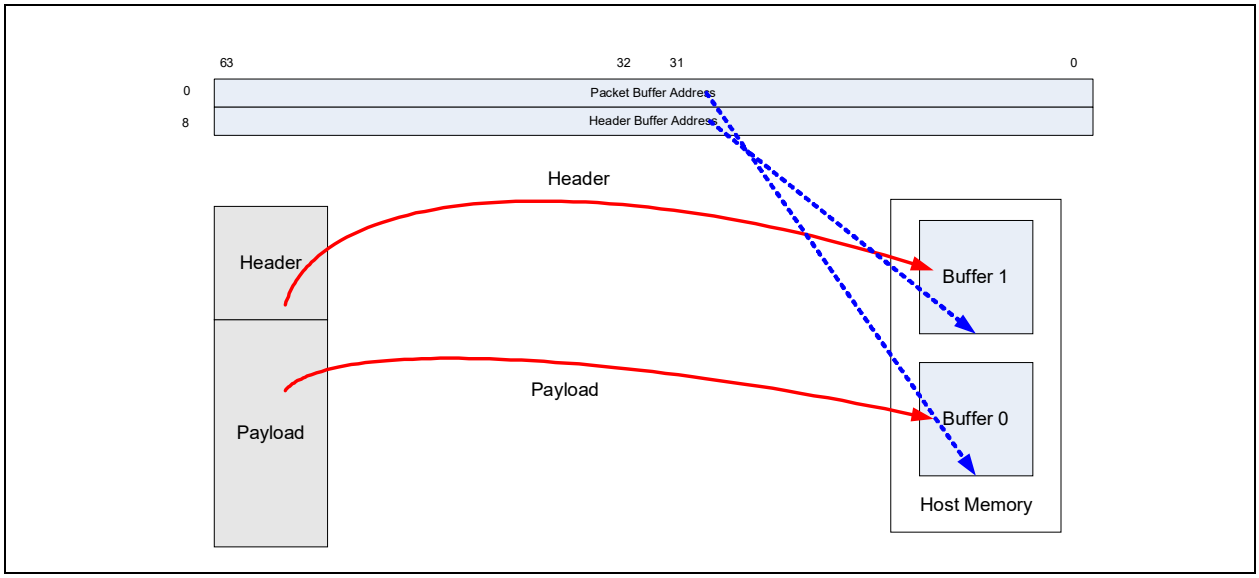


Figure 7-6. Header Splitting

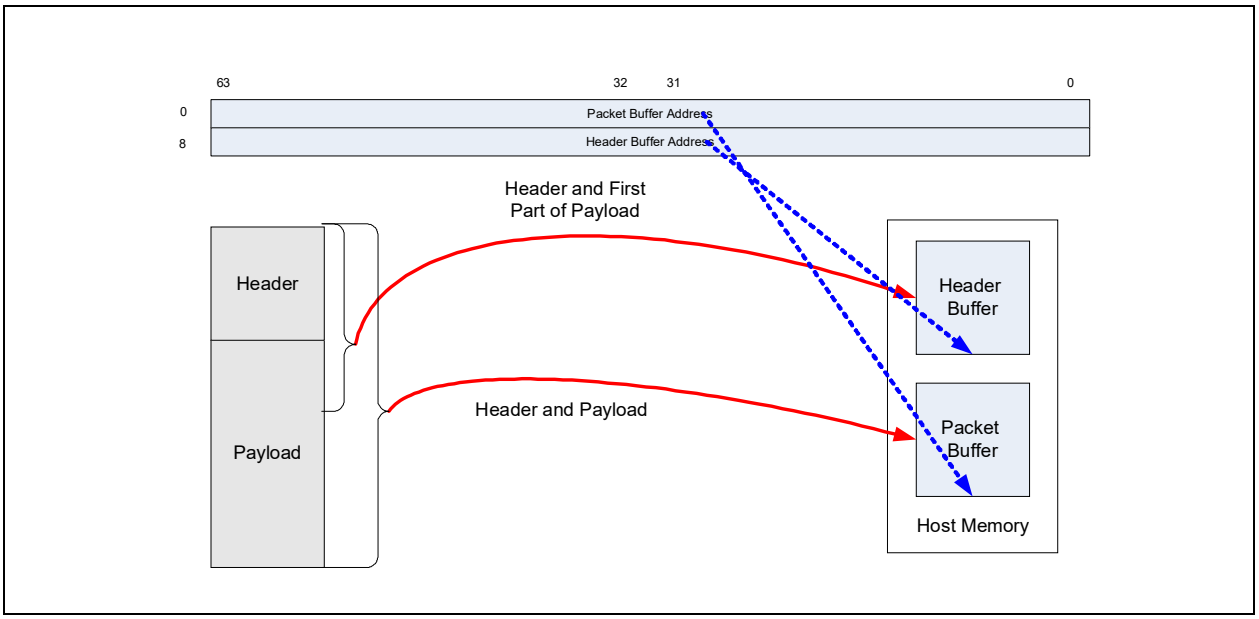


Figure 7-7. Header Replication

The physical address of each buffer is written in the *Buffer Addresses* fields. The sizes of these buffers are statically defined by *BSIZEPACKET* and *BSIZEHEADER* fields in the *SRRCTL[n]* registers.

The packet buffer address includes the address of the buffer assigned to the replicated packet, including header and data payload portions of the received packet. In the case of a split header, only the payload is included.

The header buffer address includes the address of the buffer that contains the header information. The receive DMA module stores the header portion of the received packets into this buffer.

Foxville uses the packet replication or splitting feature when the *SRRCTL[n].DESCTYPE* is larger than one. The software device driver must also program the buffer sizes in the *SRRCTL[n]* registers.

When header split is selected, the packet is split only on selected types of packets. A bit exists for each option in *PSRTYPE[n]* registers so several options can be used in conjunction with them. If one or more bits are set, the splitting is performed for the corresponding packet type. (See Appendix N.1 for details on the possible headers type supported).

Table 7-18 lists the behavior of Foxville in the different modes.

Table 7-18. Foxville Split/Replicated Header Behavior

DESCTYPE	Condition	SPH	HBO	PKT_LEN	HDR_LEN	Header and Payload DMA
Split	1. Header can't be decoded	0b	0b	Min(Packet length, *BSIZEPACKET*)	N/A	Header + Payload ⇨ Packet buffer
	2. Header <= *BSIZEHEADER*	1b	0b	Min(Payload length, *BSIZEPACKET*)[1]	Header size	Header ⇨ Header buffer Payload ⇨ Packet buffer
	3. Header > *BSIZEHEADER*	1b	1b	Min(Packet length, *BSIZEPACKET*)	Header size[2]	Header + Payload ⇨ Packet buffer

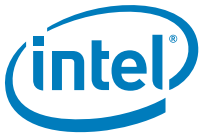
Table 7-18. Foxville Split/Replicated Header Behavior

DESCTYPE	Condition	SPH	HBO	PKT_LEN	HDR_LEN	Header and Payload DMA
Split – always use header buffer	1. Packet length <= *BSIZEHEADER*	0b	0b	Zero	Packet length	Header + Payload ⇨ Header buffer
	2. Header can't be Decoded and Packet length > *BSIZEHEADER*	0b	0b	Min(Packet length – *BSIZEHEADER*, *BSIZEPACKET*)	BSIZEHEADER	Header + Payload ⇨ Header + Packet buffers[3]
	3. Header <= *BSIZEHEADER* and Packet length >= *BSIZEHEADER*	1b	0b	Min(Payload length, *BSIZEPACKET*)	Header Size	Header ⇨ Header buffer Payload ⇨ Packet buffer
	4. Header > *BSIZEHEADER*	1b	1b	Min(Packet length – *BSIZEHEADER*, *BSIZEPACKET*)	Header Size[2]	Header + Payload ⇨ Header + Packet buffer[3]
Replicate	1. Header can't be decoded	0b[4]	0b	Min(Packet length, *BSIZEPACKET*)	N/A	(Header + Payload) (partial[5]) ⇨ Header buffer Header + Payload ⇨ Packet buffer
	2. Packet length <= *BSIZEHEADER*	1b[4]	0b	Min(Packet length, *BSIZEPACKET*)	Header size	Header + Payload ⇨ Header buffer Header + Payload ⇨ Packet buffer
	3. Packet length > *BSIZEHEADER*	1b[4]	0b/1b[5]	Min(Packet length, *BSIZEPACKET*)	Header size	Header + Payload (partial[6]) ⇨ Header buffer Header + Payload ⇨ Packet buffer
Replicate Large Packet only	1. Header can't be decoded	0b[4]	0b	Min(Packet length, *BSIZEPACKET*)	N/A	(Header + Payload) (partial[5]) ⇨ Header buffer Header + Payload ⇨ Packet buffer
	2. Packet length <= *BSIZEHEADER*	1b[4]	0b	Packet length	Header size	Header + Payload ⇨ Header buffer
	2. Packet length > *BSIZEHEADER*	1b[4]	0b/1b[5]	Min(Packet length, *BSIZEPACKET*)	Header size	(Header + Payload) (partial[5]) ⇨ Header buffer Header + Payload ⇨ Packet buffer

1. In a header only packet (such as TCP ACK packet), the PKT_LEN is zero.
2. The HDR_LEN doesn't reflect the actual data size stored in the Header buffer. It reflects the header size determined by the parser. When timestamp in packet is enabled header size reflects the additional 16 bytes of the timestamp.
3. If the packet spans more than one descriptor, only the header buffer of the first descriptor is used. The header buffer is used for the first part of the packet until it is filled up, and then the first packet buffer is used for the continuation of the packet.
4. In replicate mode if *SPH* = 0b due to no match to any of the headers selected in the *PSRTYPE[n]* register, then the header size is not relevant. In any case, even if *SPH* = 1b due to match to one of the headers selected in the *PSRTYPE[n]* register, the *HDR_LEN* doesn't reflect the actual data size stored in the header buffer.
5. HBO is 1b if the header size is bigger than *BSIZEHEADER* and zero otherwise.
6. Partial means up to *BSIZEHEADER.*

Software Notes:

- If *SRRCTL[n].NSE* is set, all buffers' addresses in a packet descriptor must be word aligned.
- Packet header can't span across buffers, therefore, the size of the header buffer must be larger than any expected header size. Otherwise, only the part of the header fitting the header buffer is replicated. In the case of header split mode (*SRRCTL[n].DESCTYPE* = 010b), a packet with a header larger than the header buffer is not split.
- Section 7.1.5 describes the details of the split/replicate conditions for different types of headers according to the settings of the *PSRTYPE* register values.



7.1.6 Receive Packet Timestamp in Buffer

Foxville supports adding an optional tailored header before the MAC header of the packet in the receive buffer. The 128 bit tailored header include a timestamp composed of the packet reception time measured in the *SYSTIML* (Low DW) and *SYSTIMH* (High DW) registers of two selected 1588 timers.

The timestamp information is placed in Host order (Little Endian) format as listed in Table 7-19. The timer '0' and time '1' in this table are selected by the following setting options:

- The 1588 timer reported as Timer '1' for queue 'n' is selected by the SRRCTL[n].Timer1_Sel field .
- The 1588 timer reported as Timer '0' for queue 'n' is selected by the SRRCTL[n].Timer0_Sel field and the DOM2TIMER register as follow.
  - If the packet is recognized as 1588 V2 event message (Foxville identifies the 1588 domain field in the packet), and the 1588 domain is defined for the queue in the DOM2TIMER register, and the SRRCTL[n].USE_DOMAIN flag is active, then the reported 1588 timer is selected by the DOM2TIMER register.
  - Else, the reported timer is selected by the Timer0_Sel field.

Note: Foxville identifies 1588 V2 event message by the following criteria:
- Packet is identified as 1588 packet version 2
- The message type field (at byte offset 0 of the 1588 message) equals to 0x0...0x7

Table 7-19. Timestamp Layout in Buffer

0 3	4 7	8 11	12 15	16...
Timer '1' - SYSTIML	Timer '1' - SYSTIMH	Timer '0' - SYSTIML	Timer '0' - SYSTIMH	Received Packet

The Timestamp in Buffer is enabled by the following settings:

- The 1588 logic must not be disabled by the *TSAUXC.Disable systime* flag (it should be cleared)
- The RXPBSIZE.CFG_TS_EN flag should be set, allocating the extra 16 bytes in the packet buffer for the received packets. This is a static setting that should be made when receive is not enabled yet.
- Specific setting of the relevant receive queues by the *SRRCTL[n] registers*
  - The *Timestamp flag should be set, instructing the hardware to post the timestamp in the packet buffer*
  - If the *DESCTYPE is set to any of the header split modes then the BSIZEHEADER field should be set to a larger header buffer than 128 bytes*

Packets are received to the queue are time stamped if they meet the criteria listed in Table 7-73 in Section 7.5.5. Meeting these cases the packet is reported as follow:

- Place a 64 bit timestamp, indicating the time a packet was received (sampled by both 1588 timers), at the beginning of the receive buffer before the received packet.
- Set the *TSIP* bit in the *RDESC.STATUS* field of the last receive descriptor of the packet.
- Update the *RDESC.Packet Type* field in the last receive descriptor (as any other packet). Value in this field enables identifying that this is a PTP (Precision Time Protocol) packet (this indication is only relevant for L2 packets).
- Update the *RDESC.HDR_LEN* and *RDESC.PKT_LEN* values to include size of timestamp.

Software driver should take into account the additional size of the timestamp when preparing the receive descriptors for the relevant queue.

7.1.7 Receive Packet Checksum and SCTP CRC Offloading

Foxville supports the off loading of the following receive integrity checks: packet Ethernet CRC, IPv4 header checksum, TCP/UDP checksum, fragment payload checksum, and SCTP CRC32.

The packet checksum and the fragment payload checksum shares the same location as the RSS field and is reported in the receive descriptor when the *RXCSUM.PCSD* bit is cleared. If the *RXCSUM.IPPCSE* is set, the Packet checksum is aimed to accelerate checksum calculation of fragmented UDP packets. Please refer to Section 7.1.7.2 for a detailed explanation. If *RXCSUM.IPPCSE* is cleared (the default value), the checksum calculation that is reported in the Rx Packet checksum field is the unadjusted 16-bit one's complement of the packet.

The packet checksum is the 16-bit one's complement of the received packet, starting from the byte indicated by *RXCSUM.PCSS* (zero corresponds to the first byte of the packet), after stripping. For packets with a VLAN header, the packet checksum includes the header if VLAN striping is not enabled by the *CTRL.VME*. If a VLAN header strip is enabled, the packet checksum and the starting offset of the packet checksum exclude the VLAN header due to masking of VLAN header. For example, for an Ethernet II frame encapsulated as an 802.3ac VLAN packet and *CTRL.VME* is set and with *RXCSUM.PCSS* set to 14, the packet checksum would include the entire encapsulated frame, excluding the 14-byte Ethernet header (DA, SA, type/length) and the 4-byte q-tag. The packet checksum does not include the Ethernet CRC if the *RCTL.SECRC* bit is set.

Software must make the required offsetting computation (to remove the bytes that should not have been included and to include the pseudo-header) prior to comparing the packet checksum against the TCP checksum stored in the packet.

Note: The *RXCSUM.PCSS* value should point to a field that is before or equal to the IP header start. Otherwise the IP header checksum or TCP/UDP checksum is not calculated correctly.

For supported packet/frame types, the entire checksum calculation can be off loaded to Foxville. If *RXCSUM.IPOFLD* is set to 1b, Foxville calculates the IPv4 checksum and indicates a pass/fail indication to software via the IPv4 *Checksum Error* bit (*RDESC.IPE*) in the *Error* field of the receive descriptor. Similarly, if *RXCSUM.TUOFLD* is set to 1b, Foxville calculates the TCP or UDP checksum and indicates a pass/fail condition to software via the TCP/UDP *Checksum Error* bit (*RDESC.L4E*). These error bits are valid when the respective status bits indicate the checksum was calculated for the packet (*RDESC.IPCS* and *RDESC.L4CS*, respectively). Similarly, if *RFCTL.IPv6_DIS* and *RFCTL.IPv6Xsum_DIS* are cleared to 0b and *RXCSUM.TUOFLD* is set to 1b, Foxville calculates the TCP or UDP checksum for IPv6 packets. It then indicates a pass/fail condition in the TCP/UDP *Checksum Error* bit (*RDESC.L4E*).

If neither *RXCSUM.IPOFLD* nor *RXCSUM.TUOFLD* are set, the *Checksum Error* bits (*IPE* and *L4E*) are 0b for all packets.

Supported frame types:

- Ethernet II
- Ethernet SNAP

Table 7-20. Supported Receive Checksum Capabilities

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/ UDP Checksum Calculation	Hardware SCTP CRC Calculation
IPv4 packets.	Yes	Yes	Yes
IPv6 packets.	No (n/a)	Yes	Yes

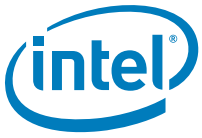


Table 7-20. Supported Receive Checksum Capabilities

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/ UDP Checksum Calculation	Hardware SCTP CRC Calculation
IPv6 packet with next header options: • Hop-by-hop options • Destinations options (without Home option) • Destinations options (with Home option) • Routing (with Segments Left zero) • Routing (with Segments Left > zero) • Fragment	 No (n/a) No (n/a) No (n/a) No (n/a) No (n/a) No (n/a)	 Yes Yes No Yes No No	Yes
IPv4 tunnels: • IPv4 packet in an IPv4 tunnel. • IPv6 packet in an IPv4 tunnel.	Yes (External - as if L3 only) Yes (IPv4)	No Yes[1]	No Yes
IPv6 tunnels: • IPv4 packet in an IPv6 tunnel. • IPv6 packet in an IPv6 tunnel.	 No No	 No No	 No No
Packet is an IPv4 fragment.	Yes	No[2]	No
Packet is greater than 1518, 1522 or 1526 bytes (LPE=1b)[3].	Yes	Yes	Yes
Packet has 802.3ac tag.	Yes	Yes	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes).	Yes	Yes	Yes
Packet has TCP or UDP options.	Yes	Yes	Yes
IP header's protocol field contains a protocol number other than TCP or UDP or SCTP.	Yes	No	No

1. The IPv6 header portion can include supported extension headers as described in the "IPv6 packet with next header options" row.
2. UDP checksum of first fragment is supported.
3. Depends on number of VLAN tags.

7.1.7.1 Filters Details

Table 7-20 lists general details about what packets are processed. In more detail, the packets are passed through a series of filters to determine if a receive checksum is calculated:

7.1.7.1.1 MAC Address Filter

This filter checks the MAC destination address to be sure it is valid (such as IA match, broadcast, multicast, etc.). The receive configuration settings determine which MAC addresses are accepted. See the various receive control configuration registers such as *RCTL* (*RCTL.UPE*, *RCTL.MPE*, *RCTL.BAM*), MTA, RAL, and RAH.

7.1.7.1.2 SNAP/VLAN Filter

This filter checks the next headers looking for an IP header. It is capable of decoding Ethernet II, Ethernet SNAP, and IEEE 802.3ac headers. It skips past any of these intermediate headers and looks for the IP header. The receive configuration settings determine which next headers are accepted. See the various receive control configuration registers such as *RCTL* (*RCTL.VFE*), *VET*, and *VFTA*.

7.1.7.1.3 IPv4 Filter

This filter checks for valid IPv4 headers. The version field is checked for a correct value (4).

IPv4 headers are accepted if they are any size greater than or equal to five (Dwords). If the IPv4 header is properly decoded, the IP checksum is checked for validity. The *RXCSUM.IPOFLD* bit must be set for this filter to pass.

7.1.7.1.4 IPv6 Filter

This filter checks for valid IPpv6 headers, which are a fixed size and have no checksum. The IPv6 extension headers accepted are: hop-by-hop, destination options, and routing. The maximum size next header accepted is 16 Dwords (64 bytes).

7.1.7.1.5 IPv6 Extension Headers

IPv4 and TCP provide header lengths, which enable hardware to easily navigate through these headers on packet reception for calculating checksum and CRCs, etc. For receiving IPv6 packets; however, there is no IP header length to help hardware find the packet's ULP (such as TCP or UDP) header. One or more IPv6 extension headers might exist in a packet between the basic IPv6 header and the ULP header. The hardware must skip over these extension headers to calculate the TCP or UDP checksum for received packets.

The IPv6 header length without extensions is 40 bytes. The IPv6 field *Next Header Type* indicates what type of header follows the IPv6 header at offset 40. It might be an upper layer protocol header such as TCP or UDP (*Next Header Type* of 6 or 17, respectively), or it might indicate that an extension header follows. The final extension header indicates with its *Next Header Type* field the type of ULP header for the packet.

IPv6 extension headers have a specified order. However, destinations must be able to process these headers in any order. Also, IPv6 (or IPv4) might be tunneled using IPv6, and thus another IPv6 (or IPv4) header and potentially its extension headers might be found after the extension headers.

The IPv4 *Next Header Type* is at byte offset nine. In IPv6, the first *Next Header Type* is at byte offset six.

All IPv6 extension headers have the *Next Header Type* in their first eight bits. Most have the length in the second eight bits (Offset Byte[1]) as listed in Table 7-21:

Table 7-21. Typical IPv6 Extended Header Format (Traditional Representation)

0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Next Header Type	Length	

Table 7-22 lists the encoding of the *Next Header Type* field and information on determining each header type's length. The IPv6 extension headers are not otherwise processed by Foxville so their details are not covered here.

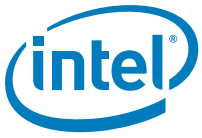


Table 7-22. Header Type Encoding and Lengths

Header	Next Header Type	Header Length (Units are Bytes Unless Otherwise Specified)
IPv6	6	Always 40 bytes
IPv4	4	Offset Bits[7:4] Unit = 4 bytes
TCP	6	Offset Byte[12].Bits[7:4] Unit = 4 bytes
UDP	17	Always 8 bytes
Hop by Hop Options	0 (Note 1)	8+Offset Byte[1]
Destination Options	60	8+Offset Byte[1]
Routing	43	8+Offset Byte[1]
Fragment	44	Always 8 bytes
Authentication	51	8+4*(Offset Byte[1])
Encapsulating Security Payload	50	Note 3
No Next Header	59	Note 2

Notes:
1. Hop-by-hop options header is only found in the first *Next Header Type* of an IPv6 header.
2. When a *No Next Header* type is encountered, the rest of the packet should not be processed.
3. Encapsulated security payload - Foxville cannot offload packets with this header type.

Note that Foxville hardware acceleration does not support all IPv6 extension header types (refer to Table 7-20).

Also, the *RFCTL.IPv6_DIS* bit must be cleared for this filter to pass.

7.1.7.1.6 UDP/TCP Filter

This filter checks for a valid UDP or TCP header. The prototype next header values are 0x11 and 0x06, respectively. The *RXCSUM.TUOFLD* bit must be set for this filter to pass.

7.1.7.2 Receive UDP Fragmentation Checksum

Foxville might provide receive fragmented UDP checksum offload. Foxville should be configured in the following manner to enable this mode:

The *RXCSUM.PCSD* bit should be cleared. The *Fragment Checksu*m and *IP Identification* fields are mutually exclusive with the RSS hash. When the RXCSUM.*PCSD* bit is cleared, *Fragment Checksum* and *IP Identification* are active instead of RSS hash.

The *RXCSUM.IPPCSE* bit should be set. This field enables the IP payload checksum enable that is designed for the fragmented UDP checksum.

The *RXCSUM.PCSS* field must be zero. The packet checksum start should be zero to enable auto-start of the checksum calculation. Table 7-23 lists the exact description of the checksum calculation.

Table 7-23 also lists the outcome descriptor fields for the following incoming packets types:

Table 7-23. Descriptor Fields

Incoming Packet Type	Fragment Checksum (if RXCSUM.PCSD is cleared)	UDPV	UDPCS / L4CS /L4I
Non IP Packet	Packet checksum	0b	0b / 0b /0b
IPv6 Packet	Packet checksum	0b	Depends on transport header.
Non fragmented IPv4 packet	Packet checksum	0b	Depends on transport header.
Fragmented IPv4, when not first fragment	The unadjusted one's complement checksum of the IP payload.	0b	1b / 0b / 0b
Fragmented IPv4, for the first fragment	Same as above	1 if the UDP header checksum is valid (not zero)	1b / 0b / 0b

Note: When the software device driver computes the 16-bit ones complement, the sum on the incoming packets of the UDP fragments, it should expect a value of 0xFFFF. Refer to Section 7.1.7 for supported packet formats.

7.1.7.3 SCTP Offload

If a receive packet is identified as SCTP, Foxville checks the CRC32 checksum of this packet if the *RXCSUM.CRCOFL* bit is set to 1b and identifies this packet as SCTP. Software is notified on the execution of the CRC check via the *L4I* bit in the *Extended Status* field of the Rx descriptor and is notified on detection of a CRC error via the *L4E* bit in the *Extended Error* field of the RX descriptor. The detection of a SCTP packet is indicated via the *SCTP* bit in the *packet Type* field of the Rx descriptor. The following SCTP packet format is expected to enable support of the SCTP CRC check:

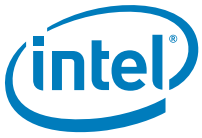
Table 7-24. SCTP Header

0 1 2 3 4 5 6 7	1 8 9 0 1 2 3 4 5	2 6 7 8 9 0 1 2 3	3 4 5 6 7 8 9 0 1
Source Port		Destination Port	
Verification Tag			
Checksum			
Chunks 1...n			

7.2 Transmit Functionality

7.2.1 Packet Transmission

Output packets to be transmitted are created using pointer-length pairs constituting a descriptor chain (descriptor based transmission). Software forms transmit packets by assembling the list of pointer-length pairs, storing this information in one of the transmit descriptor rings, and then updating the adequate on-chip transmit tail pointer. The transmit descriptors and buffers are stored in host memory. Hardware typically transmits the packet only after it has completely fetched all the packet data from host memory and stored it into the on-chip transmit FIFO (store and forward architecture). This permits TCP or UDP checksum computation and avoids problems with PCIe under-runs. Another transmit feature of Foxville is TCP/UDP segmentation. The hardware has the capability to perform packet segmentation on large data buffers offloaded from the Network Stack. This feature is discussed in detail in Section 7.2.5.



In addition, Foxville supports SCTP offloading for transmit requests. See section Section 7.2.4.3 for details about SCTP.

Table 1-11 provides a high level description of all data/control transformation steps needed for sending Ethernet packets to the line.

7.2.1.1 Transmit Data Storage

Data is stored in buffers pointed to by the descriptors. The data can be aligned to arbitrary byte boundary with the maximum size per descriptor limited only to the maximum allowed packet size (9728 bytes). A packet typically consists of two (or more) buffers, one (or more) for the header and one for the actual data. Each buffer is referenced by a different descriptor. Some software implementations might copy the header(s) and packet data into one buffer and use only one descriptor per transmitted packet.

7.2.1.2 On-Chip Transmit Buffers

Foxville allocates by default a 24 KB on-chip packet buffer. The buffers are used to store packets until they are transmitted on the line. Actual on-chip transmit buffer allocated is controlled by the *TXPBSIZE register.*

7.2.1.3 On-Chip descriptor Buffers

Foxville contains a 24 descriptor cache for each transmit queue used to reduce the latency of packet processing and to optimize the usage of the PCIe bandwidth by fetching and writing back descriptors in bursts. The fetch and write-back algorithm are described in Section 7.2.2.5 and Section 7.2.2.6.

7.2.1.4 Transmit Contexts

Foxville provides hardware checksum offload and TCP/UDP segmentation facilities. These features enable TCP and UDP packet types to be handled more efficiently by performing additional work in hardware, thus reducing the software overhead associated with preparing these packets for transmission. Part of the parameters used by these features is handled though context descriptors.

A context descriptor refers to a set of device registers loaded or accessed as a group to provide a particular function. Foxville supports 2x4 context descriptor sets (two per queue) on-chip. The transmit queues can contain transmit data descriptors (similar to the receive queue) as well as transmit context descriptors.

The contexts are queue specific and one context cannot be reused from one queue to another. This differs from the method used in previous devices that supported a pool of contexts to be shared between queues.

A transmit context descriptor differs from a data descriptor as it does not point to packet data. Instead, this descriptor provides the ability to write to the on-chip context register sets that support the transmit checksum offloading and the segmentation features of Foxville.

Foxville supports one type of transmit context. This on-chip context is written with a transmit context descriptor DTYP=2 and is always used as context for transmit data descriptor DTYP=3.

The *IDX* field contains an index to one of the two queue contexts. Software must track what context is stored in each *IDX* location.

Each advanced data descriptor that uses any of the advanced offloading features must refer to a context.

Contexts can be initialized with a transmit context descriptor and then used for a series of related transmit data descriptors. The context, for example, defines the checksum and offload capabilities for a given type of TCP/IP flow. All packets of this type can be sent using this context.

Software is responsible for ensuring that a context is only overwritten when it is no longer needed. Hardware does not include any logic to manage the on-chip contexts; it is completely up to software to populate and then use the on-chip context table.

Note: Software should not queue more than two context descriptors in sequence without an intervening data descriptor, to achieve adequate performance.

Each context defines information about the packet sent including the total size of the MAC header (*TDESC.MACLEN*), the maximum amount of payload data that should be included in each packet (*TDESC.MSS*), UDP or TCP header length (*TDESC.L4LEN*), IP header length (*TDESC.IPLEN*), and information about what type of protocol (TCP, IP, etc.) is used. Other than TCP, IP (*TDESC.TUCMD*), most information is specific to the segmentation capability.

Because there are dedicated on-chip resources for contexts, they remain constant until they are modified by another context descriptor. This means that a context can be used for multiple packets (or multiple segmentation blocks) unless a new context is loaded prior to each new packet. Depending on the environment, it might be unnecessary to load a new context for each packet. For example, if most traffic generated from a given node is standard TCP frames, this context could be setup once and used for many frames. Only when some other frame type is required would a new context need to be loaded by software. This new context could use a different index or the same index.

This same logic can also be applied to the TCP/UDP segmentation scenario, though the environment is a more restrictive one. In this scenario, the host is commonly asked to send messages of the same type, TCP/IP for instance, and these messages also have the same Maximum Segment Size (MSS). In this instance, the same context could be used for multiple TCP messages that require hardware segmentation.

7.2.2 Transmit Descriptors

Foxville supports legacy descriptors and Foxville advanced descriptors.

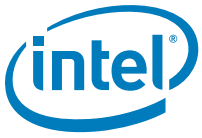
Legacy descriptors are intended to support legacy drivers to enable fast platform power up and to facilitate debug.

In addition, Foxville supports two types of advanced transmit descriptors:

1. Advanced Transmit Context Descriptor, *DTYP* = 0010b.
2. Advanced Transmit Data Descriptor, *DTYP* = 0011b.

Note: *DTYP* values 0000b and 0001b are reserved.

The transmit data descriptor (both legacy and advanced) points to a block of packet data to be transmitted. The advanced transmit context descriptor does not point to packet data. It contains control/context information that is loaded into on-chip registers that affect the processing of packets for transmission. The following sections describe the descriptor formats.



7.2.2.1 Legacy Transmit Descriptor Format

Software Note: Legacy Transmit Descriptor will not be supported by Intel any further. It is advice for new software driver programmers to use only the Advanced Transmit Descriptor.

Legacy (82542 compatible) descriptors are identified by having bit 29 of the descriptor (*TDESC.DEXT*) set to 0b. In this case, the descriptor format is defined as shown in Table 7-25. Note that the address and length must be supplied by software. Also note that bits in the command byte are optional, as is are the CSO, and CSS fields.

Table 7-25. Transmit Descriptor (TDESC) Fetch Layout - Legacy Mode

	63 48	47 40	39 36	35 32	31 24	23 16	15 0
0	Buffer Address [63:0]						
8	VLAN	CSS	Reserved	STA	CMD	CSO	Length

Table 7-26. Transmit Descriptor (TDESC) Write-Back Layout - Legacy Mode

	63 48	47 40	39 36	35 32	31 24	23 16	15 0
0	Reserved				Reserved		
8	VLAN	CSS	Reserved	STA	CMD	CSO	Length

Note: For frames that span multiple descriptors, the *VLAN, CSS, CSO, CMD.VLE, CMD.IC,* and *CMD.IFCS* are valid only in the first descriptors and are ignored in the subsequent ones.

7.2.2.1.1 Buffer Address (64)

Physical address of a data buffer in host memory that contains a portion of a transmit packet.

7.2.2.1.2 Length

Length (*TDESC.LENGTH*) specifies the length in bytes to be fetched from the buffer address provided. The maximum length associated with any single legacy descriptor is 9728 bytes.

Descriptor length(s) might be limited by the size of the transmit FIFO. All buffers comprising a single packet must be able to be stored simultaneously in the transmit FIFO. For any individual packet, the sum of the individual descriptors' lengths must be below 9728 bytes.

Note: The maximum allowable packet size for transmits is defined by the *DTXMXPKTSZ* register.

Note: Descriptors with zero length (null descriptors) transfer no data. Null descriptors can only appear between packets and must have their *EOP* bits set.

Note: If the *TCTL.PSP* bit is set, the total length of the packet transmitted, not including FCS should be at least 17 bytes. If bit is cleared the total length of the packet transmitted, not including FCS should be at least 60 bytes.

7.2.2.1.3 Checksum Offset and Start - CSO and CSS

A *Checksum Offset* (*TDESC.CSO*) field indicates where, relative to the start of the packet, to insert a TCP checksum if this mode is enabled. A *Checksum Start* (*TDESC.CSS*) field indicates where to begin computing the checksum.

Both CSO and CSS are in units of bytes and must be in the range of data provided to Foxville in the descriptors. For short packets that are not padded by software, CSS and CSO must be in the range of the unpadded data length, not the eventual padded length (64 bytes).

CSO must be set to the location of TCP checksum in the packet. CSS must be set to the beginning of the IP header or the L4 (TCP) header. Checksum calculation is not done if CSO or CSS are out of range. This occurs if (CSS > length) OR (CSO > length - 1).

In the case of an 802.1Q header, the offset values depend on the VLAN insertion enable (*VLE*) bit. If it is not set (VLAN tagging included in the packet buffers), the offset values should include the VLAN tagging. If this bit is set (VLAN tagging is taken from the packet descriptor), the offset values should exclude the VLAN tagging.

Note: Assuming CSS points to the beginning of the IP header, software must compute an offsetting entry to back out the bytes of the header that are not part of the IP pseudo header and should not be included in the TCP checksum and store it in the position where the hardware computed checksum is to be inserted. Hardware does not add the 802.1Q Ethertype or the VLAN field following the 802.1Q Ethertype to the checksum. So for VLAN packets, software can compute the values to back out only the encapsulated IP header packet and not the added fields.

Note: UDP checksum calculation is not supported by the legacy descriptors. When using legacy descriptors Foxville is not aware of the L4 type of the packet and thus, does not support the translation of a checksum result of 0x0000 to 0xFFFF needed to differentiate between an UDP packet with a checksum of zero and an UDP packet without checksum.

Because the *CSO* field is eight bits wide, it puts a limit on the location of the checksum to 255 bytes from the beginning of the packet.

Hardware adds the checksum to the field at the offset indicated by the *CSO* field. Checksum calculations are for the entire packet starting at the byte indicated by the *CSS* field. A value of zero corresponds to the first byte in the packet.

CSS must be set in the first descriptor of the packet.

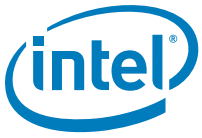
7.2.2.1.4 Command Byte - CMD

Table 7-27. Transmit Command (TDESC.CMD) Layout

7	6	5	4	3	2	1	0
RSV	VLE	DEXT	Rsv	RS	IC	IFCS	EOP

The CMD byte stores the applicable command and has the fields shown in Table 7-27.

- RSV (bit 7) - Reserved
- VLE (bit 6) - VLAN Insertion Enable (See Table 7-28).
- DEXT (bit 5) - Descriptor Extension (0 for legacy mode)
- Reserved (bit 4) - Reserved



- RS (bit 3) - Report Status
- IC (bit 2) - Insert Checksum
- IFCS (bit 1) - Insert FCS
- EOP (bit 0) - End of Packet

VLE: Indicates that the packet is a VLAN packet. For example, hardware should add the VLAN EtherType and an 802.1Q VLAN tag to the packet.

Table 7-28. VLAN Tag Insertion Decision Table

VLE	Action
0b	Send generic Ethernet packet.
1b	Send 802.1Q packet; VLAN data comes from the *VLAN* field of the TX descriptor.

RS: Signals the hardware to report the status information. This is used by software that does in-memory checks of the transmit descriptors to determine which ones are done. For example, if software queues up 10 packets to transmit, it can set the *RS* bit in the last descriptor of the last packet. If software maintains a list of descriptors with the *RS* bit set, it can look at them to determine if all packets up to (and including) the one with the *RS* bit set have been buffered in the output FIFO. Looking at the status byte and checking the *Descriptor Done* (*DD*) bit enables this operation. If *DD* is set, the descriptor has been processed. Refer to Table 7-29 for the layout of the status field.

IC: If set, requests hardware to add the checksum of the data from *CSS* to the end of the packet at the offset indicated by the *CSO* field.

IFCS: When set, hardware appends the MAC FCS at the end of the packet. When cleared, software should calculate the FCS for proper CRC check. There are several cases in which software must set IFCS:

- Transmitting a short packet while padding is enabled by the *TCTL.PSP* bit.
- Checksum offload is enabled by the *IC* bit in the TDESC.CMD.
- VLAN header insertion enabled by the *VLE* bit in the TDESC.CMD.

EOP: When set, indicates this is the last descriptor making up the packet. Note that more than one descriptor can be used to form a packet.

7.2.2.1.5 Status – STA

Table 7-29. Transmit Status (TDESC.STA) Layout

3	2	1	0
Reserved			DD

7.2.2.1.6 DD (Bit 0) - Descriptor Done Status

The DD bit provides the transmit status, when *RS* is set in the command: *DD* indicates that the descriptor is done and is written back after the descriptor has been processed.

Note: When head write back is enabled (*TDWBAL[n].Head_WB_En* = 1), there is no write-back of the *DD* bit to the descriptor. When using legacy Tx descriptors, Head writeback should not be enabled (*TDWBAL[n].Head_WB_En* = 0).

7.2.2.1.7 VLAN

The *VLAN* field is used to provide the 802.1Q/802.1ac tagging information. The *VLAN* field is valid only on the first descriptor of each packet when the *VLE* bit is set. The rule for VLAN tag is to use network ordering. The VLAN field is placed in the transmit descriptor in the following manner:

Table 7-30. VLAN Field (TDESC.VLAN) Layout

15 13	12	11 0
PRI	CFI	VLAN ID

- VLAN ID - the 12-bit tag indicating the VLAN group of the packet.
- Canonical Form Indication (CFI) - Set to zero for Ethernet packets.
- PRI - indicates the priority of the packet.

Note: The VLAN tag is sent in network order (also called big endian).

7.2.2.2 Advanced Transmit Context Descriptor

Table 7-31. Transmit Context Descriptor (TDESC) Layout - (Type = 0010b)

	63:62	61 32	31 16	15 9	8 0
0	RSV	LaunchTime	VLAN	MACLEN	IPLEN

	63 48	47 40	39 37	36	35 30	29	28 24	23 20	19 9	8 0
8	MSS	L4LEN	TSN_CNTX	IDX	RSV	DEXT	RSV[1]	DTYP	TUCMD	RSV[1]

1. RSV - Reserved - must be set to zero

7.2.2.2.1 IPLEN (9)

IP header length. If an offload is requested, *IPLEN* must be greater than or equal to 20 and less than or equal to 511.

7.2.2.2.2 MACLEN (7)

This field indicates the length of the MAC header. When an offload is requested (either *TSE* or *IXSM* or *TXSM* is set), *MACLEN* must be larger than or equal to 14 and less than or equal to 127. This field should include only the part of the L2 header supplied by the software device driver and not the parts added by hardware. Table 7-32 lists the value of *MACLEN* in the different cases.

Table 7-32. MACLEN Values

SNAP	Regular VLAN	External VLAN	MACLEN
No	By hardware or no VLAN	No	14
No	By hardware or no VLAN	Yes	18
No	By software	No	18
No	By software	Yes	22
Yes	By hardware or no VLAN	No	22
Yes	By hardware or no VLAN	Yes	26

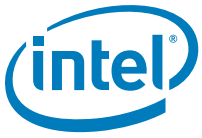


Table 7-32. MACLEN Values

SNAP	Regular VLAN	External VLAN	MACLEN
Yes	By software	No	26
Yes	By software	Yes	30

VLAN (16) - 802.1Q VLAN tag to be inserted in the packet during transmission. This VLAN tag is inserted and needed only when a packet using this context has its *DCMD.VLE* bit set. This field should include the entire 16-bit *VLAN* field including the *CFI* and *Priority* fields as listed in Table 7-30.

Note: The VLAN tag is sent in network order.

7.2.2.2.3 LaunchTime (30)

The LaunchTime is a relative offset, to the BaseT register and StQT[n] register of the queue. It defines the **scheduling time** of the packet from the packet buffer to the MAC. On top of it, the GTxOffset register is used to compensate for the latency between the scheduling "point" and the PHY MDI pins (see Table 7-62 for this latency). The packet is scheduled for transmission when the SYSTIM registers that is defined for transmit scheduling (by the Sch_Timer_Sel field in the TQAVCTRL register) is larger than the "**Scheduling Time**". So...

Transmit Time on the PHY MDI pins	= LaunchTime + BaseT + StQT
Scheduling Time	= LaunchTime + BaseT + StQT - **GTxOffset**

The LaunchTime field is active only for transmit queues on which the TXQCTL.QUEUE_MODE parameter is set to LaunchT.

The packet data is fetched from host memory either as soon as possible or as late as possible depending on the TXQCTL.DATA_FETCH_TIM parameter (see Section 7.5.2.5 for details).

Note that the transmit scheduler operates on a 156.25 MHz clock which dictates its accuracy of 6.4nsec.

7.2.2.2.4 TSN_CNTX (3)

Table 7-33. TSN_CNTX Layout

2	1	0
Frst	Reserved	

- **Frst** (bit 2) - The Frst flag indicates the first packet in a Qbv cycle. See Section 7.5.2.9.3.3 for additional details.

7.2.2.2.5 TUCMD (11)

Table 7-34. Transmit Command (TDESC.TUCMD) Layout

10 4	3 2	1	0
Reserved	L4T	IPV4	SNAP

- RSV (bit 10:4) - Reserved
- L4T (bit 3:2) - L4 Packet TYPE (00b: UDP; 01b: TCP; 10b: SCTP; 11b: Reserved)

- IPV4 (bit 1) - IP Packet Type: When 1b, IPv4; when 0b, IPv6
- SNAP (bit 0) - SNAP indication

7.2.2.2.6 DTYP(4)

Equals to 0010b for this type of descriptor.

7.2.2.2.7 DEXT(1)

Descriptor Extension (1b for advanced mode).

7.2.2.2.8 IDX (1)

Index into one of two hardware context per queue to be configured by this descriptor.

Note: For TSN queues a valid context descriptor should be placed ahead of any timed packet pointed by a data descriptor and the IDX field is ignored.

7.2.2.2.9 L4LEN (8)

Layer 4 header length. If *TSE* is set in the data descriptor pointing to this context, this field must be greater than or equal to 12 and less than or equal to 255. Otherwise, this field is ignored.

7.2.2.2.10 MSS (16)

Controls the Maximum Segment Size (MSS). It is meaningful only when the *DCMD.TSE bit in the data descriptor is set.* This specifies the maximum TCP payload segment sent per frame, not including any header or trailer. The total length of each frame (or section) sent by the TCP/UDP segmentation mechanism (excluding Ethernet CRC) as follows:

MACLEN + 4(if VLE set) + IPLEN + L4LEN + MSS

The one exception is the last packet of a TCP/UDP segmentation, which can be shorter.

Note: The headers lengths must meet the following rule: MACLEN + IPLEN + L4LEN <= 512

Note: The MSS value should be larger or equal to 64 and the maximum MSS value should not exceed 9216 bytes (9KB) length.

The context descriptor requires valid data only in the fields used by the specific offload options. Table 7-35 lists the required valid fields according to the different offload options.

Table 7-35. Valid Field in Context vs. Required Offload

Required Offload			Valid Fields in Context						
TSE	TXSM	IXSM	VLAN[1]	L4LEN	IPLEN	MACLEN	MSS	L4T	IPV4
1b[2]	1b	X[3]	VLE	Yes	Yes	Yes	Yes	Yes	Yes

Table 7-35. Valid Field in Context vs. Required Offload

0b	1b	X[2]	VLE	No	Yes	Yes	No	Yes	Yes
0b	0b	1b	VLE	No	Yes	Yes	No	No	Yes
0b	0b	0b	No context required unless VLE is set.						

1. VLAN field is required only if the *VLE* bit in Tx descriptor is set.
2. If *TSE* is set, *TXSM* must be set to 1b.
3. X - don't care.

7.2.2.3 Advanced Transmit Data Descriptor

Table 7-36. Advanced Transmit Data Descriptor (TDESD) Layout - (Type = 0011b)

0	Address[63:0]							
8	PAYLEN	POPTS	PTP2	IDX	STATUS	DCMD	DTYP	PTP1	DTALEN
	63 46	**45 40**	**39 37**	**36**	**35 32**	**31 24**	**23 20**	**19 16**	**15 0**

Table 7-37. Advanced Tx Descriptor Write-back Format

0	DMA Time Stamp		
8	Reserved	STA	Reserved
	63 36	**35 32**	**31 0**

Note: For frames that span multiple descriptors, all fields apart from *DCMD.EOP*, *DCMD.RS*, *DCMD.DEXT*, *DTALEN*, Address and DTYP are valid only in the first descriptor and are ignored in the subsequent ones.

7.2.2.3.1 Address (64) / DMA Time Stamp

Address: Physical address of a data buffer in host memory that contains a portion of a transmit packet provided by the software.

DMA Time Stamp: When enabled by the *1588_STAT_EN* flag in the *TQAVCTRL* register, the DMA Time Stamp is valid and the TS_STAT flag in the STA field is set. Otherwise, this field is undefined and the TS_STAT flag in the STA field is cleared. The DMA Time Stamp reports the time on which the descriptor is written back to host memory. In order to minimize the time gap between DMA completion and descriptor write back, the software could either use the *RS* bit or set the *WTHRESH* parameter in the *TXDCTL[n]* register (of the queue) to zero. The DMA Time Stamp reports only part of the time (in the *SYSTIM* registers) as follows. Therefore, the software should read the *SYSTIMH* register (once every ~512 sec) in order to keep track of the complete time. Each transmit queue is associated to one of the 4 x 1588 timers as defined by TXQCTL.Sch_TIMER_SEL. The reported DMA Time Stamp is provided by the associated 1588 timer of the queue.

- *DMA Time Stamp* bits 31:0 get the value of *SYSTIML* register
- *DMA Time Stamp* bits 41:32 get the value of *the 10 LS bits of the SYSTIMH* register
- *DMA Time Stamp* bits 63:42 are set to zero

7.2.2.3.2 DTALEN (16)

Length in bytes of data buffer at the address pointed to by this specific descriptor.

Note: If the *TCTL.PSP* bit is set, the total length of the packet transmitted, not including FCS, should be at least 17 bytes. If bit is cleared the total length of the packet transmitted, not including FCS should be at least 60 bytes.

Note: The maximum allowable packet size for transmits is based on the value written to the *DMA TX Max Allowable packet size* (*DTXMXPKTSZ*) register. Default value is 9,728 bytes.

7.2.2.3.3 PTP1 (4)

Sample the transmission time of the packet as defined by the PTP1 and PTP2 fields.

Table 7-38. PTP1 Field Layout

3	2	1	0
Sample_TimeStamp	1STEP_1588	Reg_Sel	

- Sample_TimeStamp (bit 3) - This flag enables the packet time stamping by one of the registers indicated in the Reg_Sel field.
- 1STEP_1588 (bit 2) - Sample the transmitted packet Time-stamp and post it in the transmitted packet at the offset defined by *1588_Offset* fields in the *TSYNCTXCTL* register. Using this option Foxville calculates the Ethernet CRC including the inserted time-stamp. Note that when transmitting the 1588 packets over UDP, Foxville does not re-calculate the UDP checksum. Therefore, UDP checksum should not by used with 1STEP option.
- Reg_Sel (bits 0:1) - Sample the transmitted packet Time-stamp to one of the 4 x TXSTMP registers defined by the Reg_Sel field.

7.2.2.3.4 PTP2 (3)

Table 7-39. Transmit Data (TDESD.MAC) Layout

2	1	0
Reserved	Timer Sel	

- Timer Sel (bit 1:0) - Selects one of the 4 x 1588 timers that should be used for the packet time stamping. This field is meaningful only if either the 1STEP_1588 or the Sample_TimeStamp flags in the PTP1 field are active.

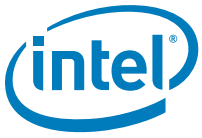
7.2.2.3.5 DTYP (4)

0011b is the value for this descriptor type.

7.2.2.3.6 DCMD (8)

Table 7-40. Transmit Data (TDESD.DCMD) Layout

7	6	5	4	3	2	1	0
TSE	VLE	DEXT	Reserved	RS	Reserved	IFCS	EOP



- TSE (bit 7) - TCP/UDP Segmentation Enable
- VLE (bit 6) - VLAN Packet Enable
- DEXT (bit 5) - Descriptor Extension (1b for advanced mode)
- Reserved (bit 4)
- RS (bit 3) - Report Status
- Reserved (bit 2)
- IFCS (bit 1) - Insert FCS
- EOP (bit 0) - End Of Packet

TSE indicates a TCP/UDP segmentation request. When *TSE* is set in the first descriptor of a TCP packet, hardware must use the corresponding context descriptor in order to perform TCP segmentation. The type of segmentation applied is defined according to the *TUCMD.L4T* field in the context descriptor.

Note: It is recommended that *TCTL.PSP* be enabled when *TSE* is used since the last frame can be shorter than 60 bytes - resulting in a bad frame if TCTL.*PSP* is disabled.

Note: Segmentation offload should not be enabled from queues on which the launch time is valid (defined by the QUEUE_MODE flag in the TXQCTL register of the queue).

VLE indicates that the packet is a VLAN packet and hardware must add the VLAN EtherType and an 802.1Q VLAN tag to the packet.

DEXT must be 1b to indicate advanced descriptor format (as opposed to legacy).

RS signals hardware to report the status information. This is used by software that does in-memory checks of the transmit descriptors to determine which ones are done. For example, if software queues up 10 packets to transmit, it can set the *RS* bit in the last descriptor of the last packet. If software maintains a list of descriptors with the *RS* bit set, it can look at them to determine if all packets up to (and including) the one with the *RS* bit set have been buffered in the output FIFO. Looking at the status byte and checking the *DD* bit do this. If *DD* is set, the descriptor has been processed. Refer to the next section for the layout of the status field.

Note: Descriptors with zero length transfer no data.

IFCS, when set, hardware appends the MAC FCS at the end of the packet. When cleared, software should calculate the FCS for proper CRC check. There are several cases in which the hardware changes the packet, and thus the software must set *IFCS*:

- Transmitting a short packet while padding is enabled by the *TCTL.PSP* bit.
- Checksum offload is enabled by the either the *TXSM* or *IXSM* bits in the *TDESD.POPTS* field.
- VLAN header insertion enabled by the *VLE* bit in the *TDESD.DCMD* descriptor field when the *VMVIR[n].VLANA* register field is 0.
- TCP/UDP segmentation offload enabled by *TSE* bit in the *TDESD.DCMD*.

EOP indicates whether this is the last buffer for an incoming packet.

7.2.2.3.7 STATUS (4)

- Rsv (bits 2-3) - Reserved
- TS_STAT (bit 1) - DMA Timestamp should be reported in the *DMA Time Stamp* field. It is enabled by the 1588_STAT_EN flag in the TQAVCTRL register.
- DD (bit 0) - Descriptor Done

7.2.2.3.8 IDX (1)

Index into one of two hardware context per queue to be used for this request. If no offload is required, this field is not relevant and no context needs to be initiated before the packet is sent. See Table 7-35 for details on type of transmit packet offloads that require a context reference.

7.2.2.3.9 POPTS (6)

Table 7-41. Transmit Data (TDESD.POPTS) Layout

5 4	3	2	1	0
SMD_Encoding	Reserved		TXSM	IXSM

- Reserved (bits 3:2)
- **TXSM** (bit 1) - Insert L4 Checksum. In this case, TUCMD.L4T in the context descriptor indicates whether the checksum is TCP, UDP, or SCTP. If this bit is set, the packet should contain at least a TCP header. When segmentation is activated (*DCMD.TSE* in TDESD is set), the TXSM must be set as well.
- **IXSM** (bit 0) - Insert IP Checksum. For IPv6 packets this bit must be cleared. If this bit is set, the packet should contain at least an IP header. When segmentation is activated (*DCMD.TSE* in TDESD is set), and *TUCMD.IPV4* is set in context descriptor, *the IXSM* must be set as well.
- **SMD_Encoding** (bits 5:4) - This field controls the transmitted SMD:
  00b - The device encodes the SFD/SMD autonomously.
  01b - The transmitted SMD indicates a "Verify" packet.
  10b - The transmitted SMD indicates a "Response" packet.
  11b - Reserved.

7.2.2.3.10 PAYLEN (18)

PAYLEN indicates the size (in byte units) of the data buffer(s) in host memory for transmission. In a single send packet, *PAYLEN* defines the entire packet size fetched from host memory. It does not include the fields that hardware adds such as: optional VLAN tagging, Ethernet CRC or Ethernet padding. When TCP or UDP segmentation offload is enabled (*DCMD.TSE* is set), *PAYLEN* defines the TCP/UDP payload size fetched from host memory.

Note: When a packet spreads over multiple descriptors, all the descriptor fields are only valid in the first descriptor of the packet, except for *RS*, which is always checked, *DTALEN* that reflects the size of the buffer in the current descriptor and *EOP*, which is always set at last descriptor of the series.

7.2.2.4 Transmit Descriptor Ring Structure

The transmit descriptor ring structure is shown in Figure 7-8. A set of hardware registers maintains each transmit descriptor ring in the host memory. New descriptors are added to the queue by software by writing descriptors into the circular buffer memory region and moving the tail pointer associated with that queue. The tail pointer points to one entry beyond the last hardware owned descriptor. Transmission continues up to the descriptor where head equals tail at which point the queue is empty.

Descriptors passed to hardware should not be manipulated by software until the head pointer has advanced past them.

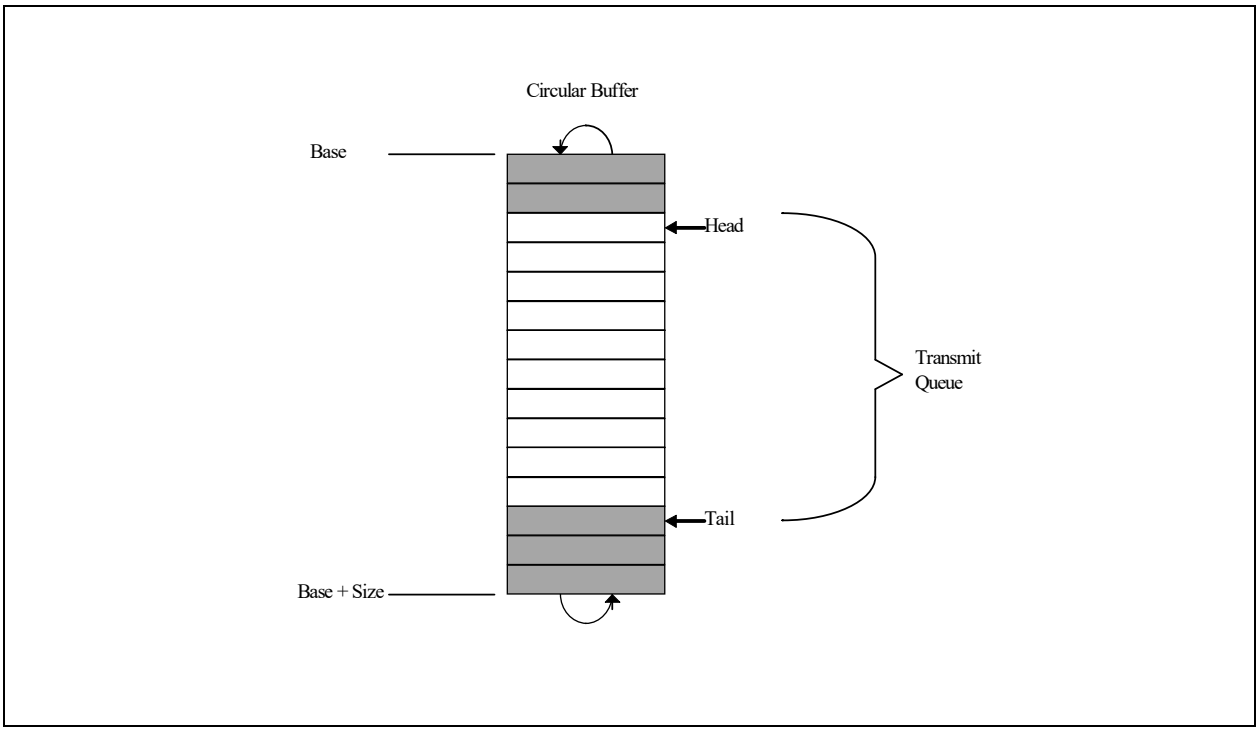


Figure 7-8. Transmit Descriptor Ring Structure

The shaded boxes in the figure represent descriptors that are not currently owned by hardware that software can modify.

The transmit descriptor ring is described by the following registers:

- Transmit Descriptor Base Address register (*TDBA 0-3*):
  This register indicates the start address of the descriptor ring buffer in the host memory; this 64-bit address is aligned on a 16-byte boundary and is stored in two consecutive 32-bit registers. Hardware ignores the lower four bits.
- Transmit Descriptor Length register (*TDLEN 0-3*):
  This register determines the number of bytes allocated to the circular buffer. This value must be zero modulo 128.
- Transmit Descriptor Head register (*TDH 0-3*):
  This register holds a value that is an offset from the base and indicates the in-progress descriptor. There can be up to 64 KB descriptors in the circular buffer. Reading this register returns the value of head corresponding to descriptors already loaded in the output FIFO. This register reflects the internal head of the hardware write-back process including the descriptor in the posted write pipe and might point further ahead than the last descriptor actually written back to the memory.
- Transmit Descriptor Tail register (*TDT 0-3*):
  This register holds a value, which is an offset from the base, and indicates the location beyond the last descriptor hardware can process. This is the location where software writes the first new descriptor.
  The driver should not handle to Foxville descriptors that describe a partial packet. Consequently, the number of descriptors used to describe a packet can not be larger than the ring size.

- Tx Descriptor Completion Write–Back Address High/Low Registers (*TDWBAH/TDWBAL* 0-3): These registers hold a value that can be used to enable operation of head write-back operation. When *TDWBAL.Head_WB_En* is set and the *RS* bit is set in the Tx descriptor, following corresponding data upload into packet buffer, Foxville writes the Transmit Descriptor Head value for this queue to the 64 bit address specified by the *TDWBAH* and *TDWBAL* registers. The Descriptor Head value is an offset from the base, and indicates the descriptor location hardware processed and software can utilize for new Transmit packets. See Section 7.2.3 for additional information.

The base register indicates the start of the circular descriptor queue and the length register indicates the maximum size of the descriptor ring. The lower seven bits of length are hard wired to 0b. Byte addresses within the descriptor buffer are computed as follows: address = base + (ptr * 16), where ptr is the value in the hardware head or tail register.

The size chosen for the head and tail registers permit a maximum of 65536 (64 KB) descriptors, or approximately 16 KB packets for the transmit queue given an average of four descriptors per packet.

Once activated, hardware fetches the descriptor indicated by the hardware head register. The hardware tail register points one descriptor beyond the last valid descriptor. Software can read and detect which packets have already been processed by hardware as follows:

- Read the head register to determine which packets (those logically before the head) have been transferred to the on-chip FIFO or transmitted. Note that this method is not recommended as races between the internal update of the head register and the actual write-back of descriptors might occur.
- Read the value of the head as stored at the address pointed by the *TDWBAH/TDWBAL* pair.
- Track the *DD* bits in the descriptor ring.

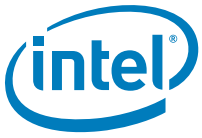
All the registers controlling the descriptor rings behavior should be set before transmit is enabled, apart from the tail registers which are used during the regular flow of data.

Note: Software can determine if a packet has been sent by either of three methods: setting the *RS* bit in the transmit descriptor command field or by performing a PIO read of the transmit head register, or by reading the head value written by Foxville to the address pointed by the *TDWBAL* and *TDWBAH* registers (see Section 7.2.3 for details). Checking the transmit descriptor *DD* bit or head value in memory eliminates a potential race condition. All descriptor data is written to the I/O bus prior to incrementing the head register, but a read of the head register could pass the data write in systems performing I/O write buffering. Updates to transmit descriptors use the same I/O write path and follow all data writes. Consequently, they are not subject to the race.

In general, hardware prefetches packet data prior to transmission. Hardware typically updates the value of the head pointer after storing data in the transmit FIFO.

7.2.2.5 Transmit Descriptor Fetching

When the *TXDCTL[n].ENABLE* bit is set and the on-chip descriptor cache is empty, a fetch happens as soon as any descriptors are made available (Host increments the *TDT[n]* tail pointer). The descriptor processing strategy for transmit descriptors is essentially the same as for receive descriptors except that a different set of thresholds are used. The number of on-chip transmit descriptors buffer per queue is 24. When there is an on-chip descriptor buffer empty, a descriptor fetch happens as soon as any descriptors are made available (host writes to the tail pointer). If several on-chip transmit descriptor queues needs to fetch descriptors, descriptors from queues that are more starved are fetched. If a number of queues have a starvation level, lowest indexed queue is served first and so forth, up to the highest indexed queue.



Note: The starvation level of a queue corresponds to the number of descriptors above the prefetch threshold (*TXDCTL[n].PTHRESH)* that are already in the internal queue. The queue is more starved if there are less descriptors in the internal transmit descriptor cache. Comparing starvation level might be done roughly, not at the single descriptor level of resolution.

A queue is considered empty for the transmit descriptor fetch algorithm as long as:

- There is still no complete packet (single or large send) in its corresponding internal queue.
- There is no descriptor already in its way from system memory to the internal cache.
- The internal corresponding cache is not full.

Each time a descriptor fetch request is sent for an empty queue, the maximum available number of descriptor is requested, regardless of cache alignment issues.

When the on-chip buffer is nearly empty (below *TXDCTL[n].PTHRESH*), a prefetch is performed each time enough valid descriptors (*TXDCTL[n].HTHRESH*) are available in host memory and no other DMA activity of greater priority is pending (descriptor fetches and write-backs or packet data transfers).

When the number of descriptors in host memory is greater than the available on-chip descriptor storage, Foxville might elect to perform a fetch that is not a multiple of cache-line size. Hardware performs this non-aligned fetch if doing so results in the next descriptor fetch being aligned on a cache-line boundary. This enables the descriptor fetch mechanism to be more efficient in the cases where it has fallen behind software.

Note: Foxville NEVER fetches descriptors beyond the descriptor tail pointer.

7.2.2.6 Transmit Descriptor Write-Back

The descriptor write-back policy for transmit descriptors is similar to that of the receive descriptors when the *TXDCTL[n].WTHRESH* value is not 0x0. In this case, all descriptors are written back regardless of the value of their *RS* bit.

When the *TXDCTL[n].WTHRESH* value is 0x0, since transmit descriptor write-backs do not happen for every descriptor, only transmit descriptors that have the *RS* bit set are written back.

Descriptors are written back in one of three cases:

- *TXDCTL[n].WTHRESH* = 0x0 and a descriptor which has *RS* set is ready to be written back.
- The corresponding *EITR* counter has reached zero. This option is meaningful only when *TXDCTL[n].WTHRESH* > 0x0.
- *TXDCTL[n].WTHRESH* > 0x0 and *TXDCTL[n].WTHRESH* descriptors have accumulated.

Note: When transmit ring is smaller than internal cache size (24 descriptors) then at least one full packet should be placed in the ring and *TXDCTL[n].WTHRESH* value should be less than ring size. If *TXDCTL[n].WTHRESH* is 0x0 (transmit *RS* mode) then at least one descriptor should have the *RS* bit set inside the ring.

7.2.3 Transmit Completions Head Write Back

In legacy hardware, transmit requests are completed by writing the *DD* bit to the transmit descriptor ring. This causes cache thrash since both the software device driver and hardware are writing to the descriptor ring in host memory. Instead of writing the *DD* bits to signal that a transmit request completed, hardware can write the contents of the descriptor queue head to host memory. The software device driver reads that memory location to determine which transmit requests are complete.

7.2.3.1 Description

The head counter is reflected in a memory location that is allocated by software, for each queue.

Head write back occurs if *TDWBAL[n].Head_WB_En* is set for this queue and the *RS* bit is set in the Tx descriptor, following corresponding data upload into packet buffer. If the head write-back feature is enabled, Foxville ignores *TXDCTL[n].WTHRESH* and takes in account only descriptors with the *RS* bit set (as if the *TXDCTL[n].WTHRESH* field was set to 0x0). In addition, the head write-back occurs upon *EITR* expiration for queues where the *WB_on_EITR* bit in *TDWBAL[n]* is set.

Software can also enable coalescing of the head write-back operations to reduce traffic on the PCIe bus, by programming the *TXDCTL.HWBTHRESH* field to a value greater than zero. In this case, head write-back operation occurs only after the internal pending write-back count is greater than the *TXDCTL[n].HWBTHRESH* value.

The software device driver has control on this feature through Tx queue 0-3 head write-back address, low (*TDWBAL[n]*) and high *(TDWBAH[n]*) registers thus supporting 64-bit address access. See registers description in Section 8.11.16 and Section 8.11.17.

The 2 low register's LSB bits of the *TDWBAL[n]* register hold the control bits.

1. The *Head_WB_En* bit enables activation of the head write back feature. When *TDWBAL[n].Head_WB_En is set to 1* no TX descriptor write-back is executed for this queue.
2. *The WB_on_EITR* bit enables head write upon *EITR* expiration. When Head write back operation is enabled (*TDWBAL[n].Head_WB_En* = 1) setting the *TDWBAL[n].WB_on_EITR* bit to 1b enables placing an upper limit on delay of head write-back operation.

The 30 upper bits of the *TDWBAL[n]* register hold the lowest 32 bits of the head write-back address, assuming that the two last bits are zero. The *TDWBAH[n]* register holds the high part of the 64-bit address.

Note: Hardware writes a full Dword when writing this value, so software should reserve enough space for each head value.

If software enables Head Write-Back, it must also disable PCI Express Relaxed Ordering on the write-back transactions. This is done by disabling bit 11 in the *TXCTL* register for each active transmit queue. See Section 8.13.2.

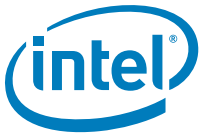
Foxville might update the Head with values that are larger then the last Head pointer, which holds a descriptor with the *RS* bit set; however, the value always points to a free descriptor (descriptor that is not longer owned by Foxville).

Note: Software should program TDWBAL[n], TDWBAH[n] registers when queue is disabled (TXDCTL[n].Enable = 0).

7.2.4 Checksum Offloading in Non-Segmentation Mode

The previous section on TCP Segmentation off-load describes the IP/TCP/UDP checksum off loading mechanism used in conjunction with TCP Segmentation. The same underlying mechanism can also be applied as a standalone feature. The main difference in normal packet mode (non-TCP Segmentation) is that only the checksum fields in the IP/TCP/UDP headers need to be updated.

Before taking advantage of Foxville's enhanced checksum off-load capability, a checksum context must be initialized. For the normal transmit checksum off-load feature this is performed by providing the device with a Descriptor with *TSE* = 0b in the *TDESD.DCMD* field and setting either the *TXSM* or *IXSM* bits in the *TDESD.POPTS* field. Setting *TSE* = 0b indicates that the normal checksum context is being set, as opposed to the segmentation context. For additional details on contexts, refer to Section 7.2.2.4.



Note: Enabling the checksum off loading capability without first initializing the appropriate checksum context leads to unpredictable results. CRC appending (*TDESC.CMD.IFCS*) must be enabled in TCP/IP checksum mode, since CRC must be inserted by hardware after the checksum has been calculated.

As mentioned in Section 7.2.2, it is not necessary to set a new context for each new packet. In many cases, the same checksum context can be used for a majority of the packet stream. In this case, some performance can be gained by only changing the context on an as needed basis or electing to use the off-load feature only for a particular traffic type, thereby avoiding the need to read all context descriptors except for the initial one.

Each checksum operates independently. Insertion of the IP and TCP checksum for each packet are enabled through the Transmit Data Descriptor *POPTS.TSXM* and *POPTS.IXSM* fields, respectively.

7.2.4.1 IP Checksum

Three fields in the Transmit Context Descriptor (*TDESC*) set the context of the IP checksum off loading feature:

- TUCMD.IPv4
- IPLEN
- MACLEN

TUCMD.IPv4 = 1b specifies that the packet type for this context is IPv4, and that the IP header checksum should be inserted. *TUCMD.IPv4* = 0b indicates that the packet type is IPv6 (or some other protocol) and that the IP header checksum should not be inserted.

MACLEN specifies the byte offset from the start of the DMA'd data to the first byte to be included in the checksum, the start of the IP header. The minimal allowed value for this field is 12. Note that the maximum value for this field is 127. This is adequate for typical applications.

Note: The *MACLEN* + *IPLEN* value needs to be less than the total DMA length for a packet. If this is not the case, the results are unpredictable.

IPLEN specifies the IP header length. Maximum allowed value for this field is 511 Bytes.

MACLEN + *IPLEN* specify where the IP checksum should stop. This is limited to the first 127 + 511 bytes of the packet and must be less than or equal to the total length of a given packet. If this is not the case, the result is unpredictable.

The 16-bit IPv4 Header Checksum is placed at the two bytes starting at *MACLEN* + 10.

As mentioned in Section 7.2.2.2, Transmit Contexts, it is not necessary to set a new context for each new packet. In many cases, the same checksum context can be used for a majority of the packet stream. In this case, some performance can be gained by only changing the context on an as needed basis or electing to use the off-load feature only for a particular traffic type, thereby avoiding all context descriptor reads except for the initial one.

7.2.4.2 TCP/UDP Checksum

Three fields in the Transmit Context Descriptor (*TDESC*) set the context of the TCP/UDP checksum off loading feature:

- MACLEN
- IPLEN

- TUCMD.L4T

TUCMD.L4T = 01b specifies that the packet type is TCP, and that the 16-bit TCP header checksum should be inserted at byte offset *MACLEN* + *IPLEN* +16. *TUCMD.L4T* = 00b indicates that the packet is UDP and that the 16-bit checksum should be inserted starting at byte offset *MACLEN* + *IPLEN* + 6.

IPLEN + *MACLEN* specifies the byte offset from the start of the DMA'd data to the first byte to be included in the checksum, the start of the TCP header. The minimal allowed value for this sum is 32/42 for UDP or TCP respectively.

Note: The *IPLEN* + *MACLEN* + *L4LEN* value needs to be less than the total DMA length for a packet. If this is not the case, the results are unpredictable.

The TCP/UDP checksum always continues to the last byte of the DMA data.

Note: For non-TSO, software still needs to calculate a full checksum for the TCP/UDP pseudo-header. This checksum of the pseudo-header should be placed in the packet data buffer at the appropriate offset for the checksum calculation.

7.2.4.3 SCTP CRC Offloading

For SCTP packets, a CRC32 checksum offload is provided.

Three fields in the Transmit Context Descriptor (*TDESC*) set the context of the STCP checksum off loading feature:

- MACLEN
- IPLEN
- TUCMD.L4T

TUCMD.L4T = 10b specifies that the packet type is SCTP, and that the 32-bit STCP CRC should be inserted at byte offset *MACLEN* + *IPLEN* + 8.

IPLEN + *MACLEN* specifies the byte offset from the start of the DMA'd data to the first byte to be included in the checksum, the start of the STCP header. The minimal allowed value for this sum is 26.

The SCTP CRC calculation always continues to the last byte of the DMA data.

The SCTP total L3 payload size (*TDESCD.PAYLEN* - *IPLEN* - *MACLEN*) should be a multiple of 4 bytes (SCTP padding not supported).

Notes:

1. TSO is not available for SCTP packets.
2. The CRC field of the SCTP header must be set to zero prior to requesting a CRC calculation offload.

7.2.4.4 Checksum Supported Per Packet Types

Table 7-42 lists which checksum is supported per packet type.

Note: TSO is not supported for packet types for which IP checksum and TCP checksum can not be calculated.

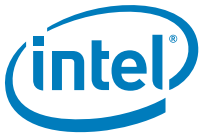


Table 7-42. Checksum per Packet Type

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP/SCTP Checksum Calculation
IPv4 packets	Yes	Yes
IPv6 packets	No (n/a)	Yes
IPv6 packet with next header options: • Hop-by-Hop options • Destinations options • Routing (w len 0b) • Routing (w len >0b) • Fragment • Home option	 No (n/a) No (n/a) No (n/a) No (n/a) No (n/a) No (n/a)	 Yes Yes Yes No No No
IPv4 tunnels: • IPv4 packet in an IPv4 tunnel • IPv6 packet in an IPv4 tunnel	 Either IP or TCP/SCTP Either IP or TCP/SCTP	 Either IP or TCP/SCTP [1] Either IP or TCP/SCTP
IPv6 tunnels: • IPv4 packet in an IPv6 tunnel • IPv6 packet in an IPv6 tunnel	 No No	 Yes Yes
Packet is an IPv4 fragment	Yes	No
Packet is greater than 1518, 1522 or 1526 bytes; (LPE=1b)[2]	Yes	Yes
Packet has 802.3ac tag	Yes	Yes
Packet has TCP or UDP options	Yes	Yes
IP header's protocol field contains protocol # other than TCP or UDP.	Yes	No

1. For the tunneled case, the driver might do only the TCP checksum or IPv4 checksum. If TCP checksum is desired, the driver should define the IP header length as the combined length of both IP headers in the packet. If an IPv4 checksum is required, the IP header length should be set to the IPv4 header length.
2. Depends on number of VLAN tags.

7.2.5 TCP/UDP Segmentation

Hardware TCP segmentation is one of the offloading options supported by the Windows* and Linux* TCP/IP stack. This is often referred to as TCP Segmentation Offloading or TSO. This feature enables the TCP/IP stack to pass to the network device driver a message to be transmitted that is bigger than the Maximum Transmission Unit (MTU) of medium. It is then the responsibility of the software device driver and hardware to divide the TCP message into MTU size frames that have appropriate layer 2 (Ethernet), 3 (IP), and 4 (TCP) headers. These headers must include sequence number, checksum fields, options and flag values as required. Note that some of these values (such as the checksum values) are unique for each packet of the TCP message and other fields such as the source IP address are constant for all packets associated with the TCP message.

Foxville supports also UDP segmentation for embedded applications, although this offload is not supported by the regular Windows* and Linux* stacks. Any reference in this section to TCP segmentation, should be considered as referring to both TCP and UDP segmentation.

The offloading of these mechanisms from the software device driver to Foxville saves significant CPU cycles. Note that the software device driver shares the additional tasks to support these options.

Note: Padding (TCTL.PSP) must be enabled in TCP segmentation mode, since the last frame might be shorter than 60 bytes, resulting in a bad frame if *PSP* is disabled.

Note: TSN must not be enabled for transmit queue on which segmentation offload is activated.

7.2.5.1 Assumptions

The following assumptions apply to the TCP segmentation implementation in Foxville:

- The *RS* bit operation is not changed.
- Interrupts are set after data in buffers pointed to by individual descriptors is transferred (DMA'd) to hardware.
- Segmentation offload should is forbidden from queues on which the launch time is valid (enabled by the QUEUE_MODE flag in the TXQCTL register of the queue).

7.2.5.2 Transmission Process

The transmission process for regular (non-TCP segmentation packets) involves:

- The protocol stack receives from an application a block of data that is to be transmitted.
- The protocol stack calculates the number of packets required to transmit this block based on the MTU size of the media and required packet headers.

For each packet of the data block:

- Ethernet, IP and TCP/UDP headers are prepared by the stack.
- The stack interfaces with the software device driver and commands it to send the individual packet.
- The software device driver gets the frame and interfaces with the hardware.
- The hardware reads the packet from host memory (via DMA transfers).
- The software device driver returns ownership of the packet to the Network Operating System (NOS) when hardware has completed the DMA transfer of the frame (indicated by an interrupt).

The transmission process for Foxville TCP segmentation offload implementation involves:

- The protocol stack receives from an application a block of data that is to be transmitted.
- The stack interfaces to the software device driver and passes the block down with the appropriate header information.
- The software device driver sets up the interface to the hardware (via descriptors) for the TCP segmentation context.

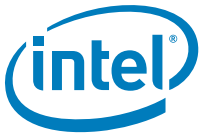
Hardware DMA's (transfers) the packet data and performs the Ethernet packet segmentation and transmission based on offset and payload length parameters in the TCP/IP context descriptor including:

- Packet encapsulation
- Header generation and field updates including IPv4, IPV6, and TCP/UDP checksum generation
- The software device driver returns ownership of the block of data to the NOS when hardware has completed the DMA transfer of the entire data block (indicated by an interrupt).

7.2.5.2.1 TCP Segmentation Data Fetch Control

To perform TCP Segmentation in Foxville, the DMA must be able to fit at least one packet of the segmented payload into available space in the on-chip Packet Buffer. The DMA does various comparisons between the remaining payload and the Packet Buffer available space, fetching additional payload and sending additional packets as space permits.

To support interleaving between descriptor queues at Ethernet frame resolution inside TSO requests, the frame header pointed to by the so called header descriptors are reread from system memory by hardware for every LSO segment. Foxville stores in an internal cache only the header's descriptors instead of the header's content.



To limit the internal cache size software should not spread the L3/L4 header (TCP, UDP, IPV4 or IPV6) on more than 4 descriptors. In the last header buffer it's allowed to mix header and data. This limitation stands for up to Layer4 header included, and for IPv4 or IPv6 indifferently.

7.2.5.2.2 TCP Segmentation Write-Back Modes

Since the TCP segmentation mode uses the buffers that contains the L3/L4 header multiple times, there are some limitations on the usage of different combinations of writeback and buffer release methods in order to guarantee the header buffer's availability until the entire packet is processed. These limitations are listed in Table 7-43.

Table 7-43. Write Back Options for Large Send

WTHRESH	RS	HEAD Write Back Enable	Hardware Behavior	Software Expected Behavior for TSO packets.
0	Set in EOP descriptors only	Disable	Hardware writes back descriptors with *RS* bit set one at a time.	Software can retake ownership of all descriptors up to last descriptor with *DD* bit set.
0	Set in any descriptors	Disable	Hardware writes back descriptors with *RS* bit set one at a time.	Software can retake ownership of entire packets (EOP bit set) up to last descriptor with *DD* bit set.
0	Not set at all	Disable	Hardware does not write back any descriptor (since *RS* bit is not set)	Software should poll the TDH register. The TDH register reflects the last descriptor that software can take ownership of.[1]
0	Not set at all	Enable	Hardware writes back the head pointer only at EITR expire event reflecting the last descriptor that software can take ownership of.	Software might poll the TDH register or use the head value written back at EITR expire event. The TDH register reflects the last descriptor that software can take ownership of.
>0	Don't care	Disable	Hardware writes back all the descriptors in bursts and set all the *DD* bits.	Software can retake ownership of entire packets up to last descriptor with both DD and EOP bits set. Note: The TDH register reflects the last descriptor that software can take ownership of[1].
Don't care	Set in EOP descriptors only	Enable	Hardware writes back the Head pointer per each descriptor with *RS* bit set.[2]	Software can retake ownership of all descriptors up to the descriptor pointed by the head pointer read from system memory (by interrupt or polling).
Don't care	Set in any descriptors	Enable	Hardware writes back the Head pointer per each descriptor with *RS* bit set.	This mode is illegal since software won't access the descriptor, it cannot tell when the pointer passed the EOP descriptor.

1. Note that polling of the TDH register is a valid method only when the *RS* bit is never set, otherwise race conditions between software and hardware accesses to the descriptor ring can occur.
2. At EITR expire event, the Hardware writes back the head pointer reflecting the last descriptor that software can take ownership of.

7.2.5.3 TCP Segmentation Performance

Performance improvements for a hardware implementation of TCP Segmentation off-load include:

- The stack does not need to partition the block to fit the MTU size, saving CPU cycles.
- The stack only computes one Ethernet, IP, and TCP header per segment, saving CPU cycles.
- The Stack interfaces with the device driver only once per block transfer, instead of once per frame.
- Larger PCIe bursts are used which improves bus efficiency (such as lowering transaction overhead).
- Interrupts are easily reduced to one per TCP message instead of one per packet.
- Fewer I/O accesses are required to command the hardware.

7.2.5.4 Packet Format

Typical TCP/IP transmit window size is 8760 bytes (about 6 full size frames). Today the average size on corporate Intranets is 12-14 KB, and normally the maximum window size allowed is 64KB (unless Windows Scaling - RFC 1323 is used). A TCP message can be as large as 256 KB and is generally fragmented across multiple pages in host memory. Foxville partitions the data packet into standard Ethernet frames prior to transmission according to the requested MSS. Foxville supports calculating the Ethernet, IP, TCP, and UDP headers, including checksum, on a frame-by-frame basis.

Table 7-44. TCP/IP or UDP/IP Packet Format Sent by Host

L2/L3/L4 Header			Data
Ethernet	IPv4/IPv6	TCP/UDP	DATA (full TCP message)

Table 7-45. TCP/IP or UDP/IP Packet Format Sent by Foxville

L2/L3/L4 Header (updated)	Data (first MSS)	FCS	...	L2/L3/L4 Header (updated)	Data (Next MSS)	FCS	...

Frame formats supported by Foxville include:

- Ethernet 802.3
- IEEE 802.1Q VLAN (Ethernet 802.3ac)
- Ethernet Type 2
- Ethernet SNAP
- IPv4 headers with options
- IPv6 headers with extensions
- TCP with options
- UDP with options.

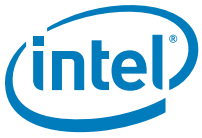
VLAN tag insertion might be handled by hardware

Note: UDP (unlike TCP) is not a “reliable protocol”, and fragmentation is not supported at the UDP level. UDP messages that are larger than the MTU size of the given network medium are normally fragmented at the IP layer. This is different from TCP, where large TCP messages can be fragmented at either the IP or TCP layers depending on the software implementation. Foxville has the ability to segment UDP traffic (in addition to TCP traffic), however, because UDP packets are generally fragmented at the IP layer, Foxville's “TCP Segmentation” feature is not normally useful to handle UDP traffic.

7.2.5.5 TCP/UDP Segmentation Indication

Software indicates a TCP/UDP Segmentation transmission context to the hardware by setting up a TCP/IP Context Transmit Descriptor (see Section 7.2.2). The purpose of this descriptor is to provide information to the hardware to be used during the TCP segmentation off-load process.

Setting the *TSE* bit in the TDESD.DCMD field to 1b indicates that this descriptor refers to the TCP Segmentation context (as opposed to the normal checksum off loading context). This causes the checksum off loading, packet length, header length, and maximum segment size parameters to be loaded from the Context descriptor into the device.



The TCP Segmentation prototype header is taken from the packet data itself. Software must identity the type of packet that is being sent (IPv4/IPv6, TCP/UDP, other), calculate appropriate checksum off loading values for the desired checksum, and calculate the length of the header which is pre-appended. The header might be up to 240 bytes in length.

Once the TCP Segmentation context has been set, the next descriptor provides the initial data to transfer. This first descriptor(s) must point to a packet of the type indicated. Furthermore, the data it points to might need to be modified by software as it serves as the prototype header for all packets within the TCP Segmentation context. The following sections describe the supported packet types and the various updates which are performed by hardware. This should be used as a guide to determine what must be modified in the original packet header to make it a suitable prototype header.

The following summarizes the fields considered by the driver for modification in constructing the prototype header.

IP Header

For IPv4 headers:

- Identification Field should be set as appropriate for first packet to be sent
- Header Checksum should be zeroed out unless some adjustment is needed by the driver

TCP Header

- Sequence Number should be set as appropriate for first packet of send (if not already)
- PSH, and FIN flags should be set as appropriate for LAST packet of send
- TCP Checksum should be set to the partial pseudo-header sum as follows (there is a more detailed discussion of this is Section 7.2.5.6):

Table 7-46. TCP Partial Pseudo-Header Sum for IPv4

IP Source Address		
IP Destination Address		
Zero	Layer 4 Protocol ID	Zero

Table 7-47. TCP Partial Pseudo-Header Sum for IPv6

IPv6 Source Address	
IPv6 Final Destination Address	
Zero	
Zero	Next Header

UDP Header

- Checksum should be set as in TCP header, above

The following sections describe the updating process performed by the hardware for each frame sent using the TCP Segmentation capability.

7.2.5.6 Transmit Checksum Offloading with TCP/UDP Segmentation

Foxville supports checksum off-loading as a component of the TCP Segmentation off-load feature and as a standalone capability. Section 7.2.4 describes the interface for controlling the checksum off-loading feature. This section describes the feature as it relates to TCP Segmentation.

Foxville supports IP and TCP header options in the checksum computation for packets that are derived from the TCP Segmentation feature.

Note: Foxville is capable of computing one level of IP header checksum and one TCP/UDP header and payload checksum. In case of multiple IP headers, the driver needs to compute all but one IP header checksum. Foxville calculates check sums on the fly on a frame-by-frame basis and inserts the result in the IP/TCP/UDP headers of each frame. TCP and UDP checksum are a result of performing the checksum on all bytes of the payload and the pseudo header.

Two specific types of checksum are supported by the hardware in the context of the TCP Segmentation off-load feature:

- IPv4 checksum
- TCP checksum

See Section 7.2.4 for description of checksum off loading of a single-send packet.

Each packet that is sent via the TCP segmentation off-load feature optionally includes the IPv4 checksum and either the TCP checksum.

All checksum calculations use a 16-bit wide one's complement checksum. The checksum word is calculated on the outgoing data.

Table 7-48. Supported Transmit Checksum Capabilities

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP Checksum Calculation
IP v4 packets	Yes	Yes
IP v6 packets (no IP checksum in IPv6)	NA	Yes
Packet is greater than 1518, 1522 or 1526 bytes; (LPE=1b)[1]	Yes	Yes
Packet has 802.3ac tag	Yes	Yes
Packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet has TCP or UDP options	Yes	Yes
IP header's protocol field contains a protocol # other than TCP or UDP.	Yes	No

1. Depends on number of VLAN tags.

Table 7-49lists the conditions of when checksum off loading can/should be calculated.

Table 7-49. Conditions for Checksum Offloading

Packet Type	IPv4	TCP/UDP	Reason
Non TSO	Yes	No	IP Raw packet (non TCP/UDP protocol)

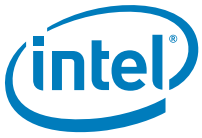


Table 7-49. Conditions for Checksum Offloading

	Yes	Yes	TCP segment or UDP datagram with checksum off-load
	No	No	Non-IP packet or checksum not offloaded
TSO	Yes	Yes	For TSO, checksum off-load must be done

7.2.5.7 TCP/UDP/IP Headers Update

IP/TCP or IP/UDP header is updated for each outgoing frame based on the IP/TCP header prototype which hardware DMA's from the first descriptor(s). The checksum fields and other header information are later updated on a frame-by-frame basis. The updating process is performed concurrently with the packet data fetch.

The following sections define what fields are modified by hardware during the TCP Segmentation process by Foxville.

Note: Placing incorrect values in the Context descriptors might cause failure of Large Send. The indication of Large Send failure can be checked in the *TSCTC* statistics register.

7.2.5.7.1 TCP/UDP/IP Headers for the First Frames

The hardware makes the following changes to the headers of the first packet that is derived from each TCP segmentation context.

MAC Header (for SNAP)

- Type/Len field = *MSS* + *MACLEN* + *IPLEN* + *L4LEN* - 14 - 4 (if VLAN added by Software)

IPv4 Header

- IP Identification: Value in the IPv4 header of the prototype header in the packet data itself
- IP Total Length = *MSS* + *L4LEN* + *IPLEN*
- IP Checksum

IPv6 Header

- Payload Length = *MSS* + *L4LEN* + IPV6_HDR_extension[1]

TCP Header

- Sequence Number: The value is the Sequence Number of the first TCP byte in this frame.
- The flag values of the first frame are set by ANDing the flag word in the pseudo header with the *DTXTCPFLGL.TCP_flg_first_seg* register field. The default value of the *DTXTCPFLGL.TCP_flg_first_seg* are set so that the FIN flag and the PSH flag are cleared in the first frame.
- TCP Checksum

UDP Header

- UDP Length = MSS + L4LEN
- UDP Checksum

7.2.5.7.2 TCP/UDP/IP Headers for the Subsequent Frames

1. IPV6_HDR_extension is calculated as IPLEN - 40 bytes.

The hardware makes the following changes to the headers for subsequent packets that are derived as part of a TCP segmentation context:

Number of bytes left for transmission = *PAYLEN* - (N * MSS). Where N is the number of frames that have been transmitted.

MAC Header (for SNAP Packets)

Type/Len field = *MSS* + *MACLEN* + *IPLEN* + *L4LEN* - 14 - 4 (if VLAN added by Software)

IPv4 Header

- IP Identification: incremented from last value (wrap around based on 16 bit-width)
- IP Total Length = *MSS* + *L4LEN* + *IPLEN*
- IP Checksum

IPv6 Header

- Payload Length = *MSS* + *L4LEN* + IPV6_HDR_extension[1]

TCP Header

- Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the *MSS* to the previous sequence number.
- The flag values of the subsequent frames are set by ANDing the flag word in the pseudo header with the *DTXTCPFLGL.TCP_Flg_mid_seg* register field. The default value of the *DTXTCPFLGL.TCP_Flg_mid_seg* are set so that if the FIN flag and the PSH flag are cleared in these frames.
- TCP Checksum

UDP Header

- UDP Length = MSS + L4LEN
- UDP Checksum

7.2.5.7.3 TCP/UDP/IP Headers for the Last Frame

The hardware makes the following changes to the headers for the last frame of a TCP segmentation context:

Last frame payload bytes = *PAYLEN* - (N * *MSS*)

MAC Header (for SNAP Packets)

- Type/Len field = Last frame payload bytes + *MACLEN* + *IPLEN* + *L4LEN* - 14 - 4 (if VLAN added by Software)

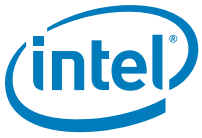
IPv4 Header

- IP Total Length = last frame payload bytes + *L4LEN* + *IPLEN*
- IP Identification: incremented from last value (wrap around based on 16 bit-width)
- IP Checksum

IPv6 Header

- Payload Length = last frame payload bytes + *L4LEN* + IPV6_HDR_extension[1]

1. IPV6_HDR_extension is calculated as IPLEN - 40 bytes.



TCP Header

- Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
- The flag values of the last frames are set by ANDing the flag word in the pseudo header with the DTXTCPFLGH.TCP_Flg_lst_seg register field. The default value of the *DTXTCPFLGH.TCP_Flg_lst_seg* are set so that if the FIN flag and the PSH flag are set in the last frame.
- TCP Checksum

UDP Header

- UDP Length = Last frame payload bytes + L4LEN
- UDP Checksum

7.2.5.8 Data Flow

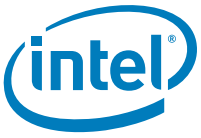
The flow used by Foxville to do TCP segmentation is as follows:

1. Get a descriptor with a request for a TSO off-load of a TCP packet.
2. First Segment processing:
   a. Fetch all the buffers containing the header as calculated by the *MACLEN*, *IPLEN* and *L4LEN* fields. Save the addresses and lengths of the buffers containing the header (up to 4 buffers). The header content is not saved.
   b. Fetch data up to the MSS from subsequent buffers & calculate the adequate checksum(s).
   c. Update the Header accordingly and update internal state of the packet (next data to fetch and TCP SN).
   d. Send the packet to the network.
   e. If total packet was sent, go to step 4. else continue.
3. Next segments
   a. Wait for next arbitration of this queue.
   b. Fetch all the buffers containing the header from the saved addresses. Subsequent reads of the header might be done with a no snoop attribute.
   c. Fetch data up to the MSS or end of packet from subsequent buffers & calculate the adequate checksum(s).
   d. Update the Header accordingly and update internal state of the packet (next data to fetch and TCP SN).
   e. If total packet was sent, request is done, else restart from step 3.
4. Release all buffers (update head pointer).

Note: Descriptors are fetched in a parallel process according to the consumption of the buffers.

7.2.6 Multiple Transmit Queues

The number of transmit queues is 4, it is useful for load balancing between CPU cores/threads as well as TSN functionality. Queue priority and arbitration scheme between the queues is described in Section 7.5.2.5. Note that throughput of low priority queues can be significantly impacted by high priority queues.



7.3 Interrupts

7.3.1 Interrupt Modes

Foxville supports the following interrupt modes:

- PCI legacy interrupts or MSI - selected when *GPIE.Multiple_MSIX* is 0b
- MSI-X when *GPIE.Multiple_MSIX* is 1b

7.3.1.1 MSI-X and Vectors

MSI-X defines a separate optional extension to basic MSI functionality. Compared to MSI, MSI-X supports a larger maximum number of vectors, the ability for software to control aliasing when fewer vectors are allocated than requested, plus the ability for each vector to use an independent address and data value, specified by a table that resides in Memory Space. However, most of the other characteristics of MSI-X are identical to those of MSI. For more information on MSI-X, refer to the PCI Local Bus Specification, Revision 3.0.

MSI-X maps each of Foxville interrupt causes into an interrupt vector that is conveyed by Foxville as a posted-write PCIe transaction. Mapping of an interrupt cause into an MSI-X vector is determined by system software (a device driver) through a translation table stored in the MSI-X Allocation registers. Each entry of the allocation registers defines the vector for a single interrupt cause.

7.3.2 Mapping of Interrupt Causes

There are 10 extended interrupt causes that exist in Foxville:

1. 8 traffic causes — 4 Tx, 4 Rx.
2. TCP timer
3. Other causes — Summarizes legacy interrupts into one extended cause.

The way Foxville exposes causes to the software is determined by the interrupt mode described in the text that follows.

Mapping of interrupts causes is different in each of the interrupt modes and is described in the following sections of this chapter.

Note: If only one MSI-X vector is allocated by the operating system, then the driver might use the non MSI-X mapping method even in MSI-X mode.

7.3.2.1 Legacy and MSI Interrupt Modes

In legacy and MSI modes, an interrupt cause is reflected by setting a bit in the *EICR* register. This section describes the mapping of interrupt causes, like a specific Rx queue event or a Link Status Change event, to bits in the *EICR* register.

Mapping of queue-related causes is accomplished through the IVAR register. Each possible queue interrupt cause (each Rx or Tx queue) is allocated an entry in the IVAR, and each entry in the IVAR identifies one bit in the EICR register among the bits allocated to queue interrupt causes. It is possible to map multiple interrupt causes into the same *EICR* bit.

In this mode, different queue related interrupt causes can be mapped to the first 4 bits of the *EICR* register.

Interrupt causes related to non-queue causes are mapped into the ICR legacy register; each cause is allocated a separate bit. The sum of all causes is reflected in the *Other Cause* bit in EICR. Figure 7-9 shows the allocation process.

The following configuration and parameters are involved:

- The IVAR[3:0] entries map 4 Tx queues and 4 Rx queues into the EICR[3:0] bits.
- The IVAR_MISC that maps non-queue causes is not used.
- The *EICR[30]* bit is allocated to the TCP timer interrupt cause.
- The *EICR[31]* bit is allocated to the other interrupt causes summarized in the ICR register.
- A single interrupt vector is provided.

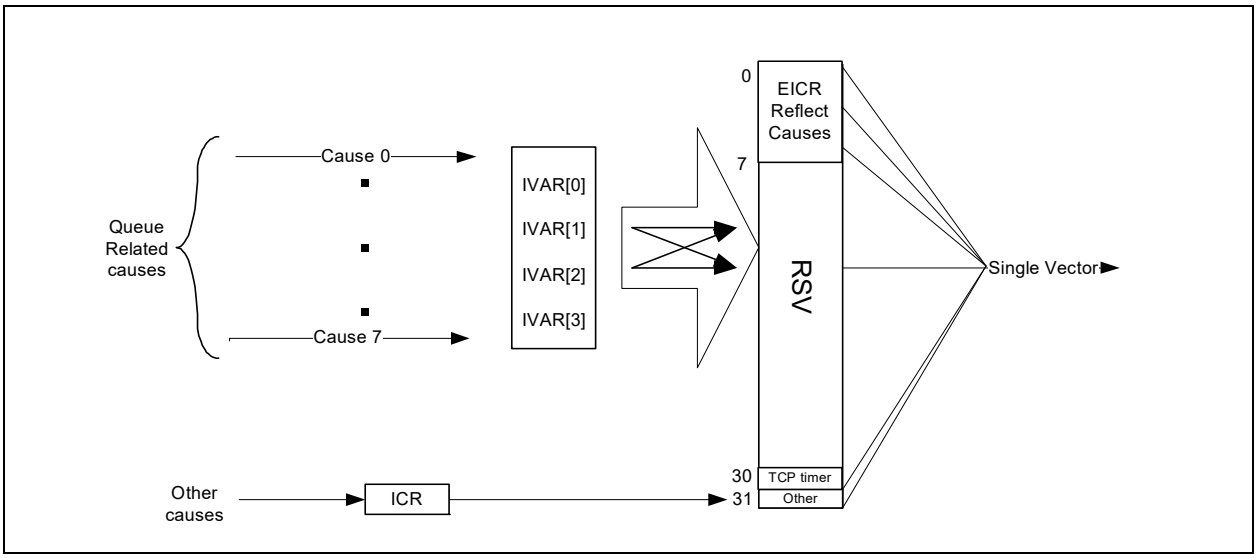


Figure 7-9. Cause Mapping in Legacy Mode

Table 7-50 lists the different interrupt causes into the IVAR registers.

Table 7-50. Cause Allocation in the IVAR Registers - MSI and Legacy Mode

Interrupt	Entry	Description
Rx_i	INT_Alloc[2*i] (i = 0..3)	Receive queues i - Associates an interrupt occurring in the Rx queue i with a corresponding bit in the EICR register.
Tx_i	INT_Alloc[2*i+1] (i = 0..3)	Transmit queues i- Associates an interrupt occurring in the Tx queue i with a corresponding bit in the EICR register.

7.3.2.2 MSI-X Mode

In MSI-X mode, Foxville can request up to 5 vectors.

In MSI-X mode, an interrupt cause is mapped into an MSI-X vector. This section describes the mapping of interrupt causes, like a specific RX queue event or a Link Status Change event, to MSI-X vectors.

Mapping is accomplished through the IVAR register. Each possible cause for an interrupt is allocated an entry in the IVAR, and each entry in the IVAR identifies one MSI-X vector. It is possible to map multiple interrupt causes into the same MSI-X vector.

The EICR also reflects interrupt vectors. The EICR bits allocated for queue causes reflect the MSI-X vector (bit 2 is set when MSI-X vector 2 is used). Interrupt causes related to non-queue causes are mapped into the ICR (as in the legacy case). The MSI-X vector for all such causes is reflected in the EICR.

The following configuration and parameters are involved:

- The IVAR[3:0] registers map 4 Tx queues and 4 Rx queues events to up to 23 interrupt vectors
- The IVAR_MISC register maps a TCP timer and other events to 2 MSI-X vectors

Figure 7-10 shows the allocation process.

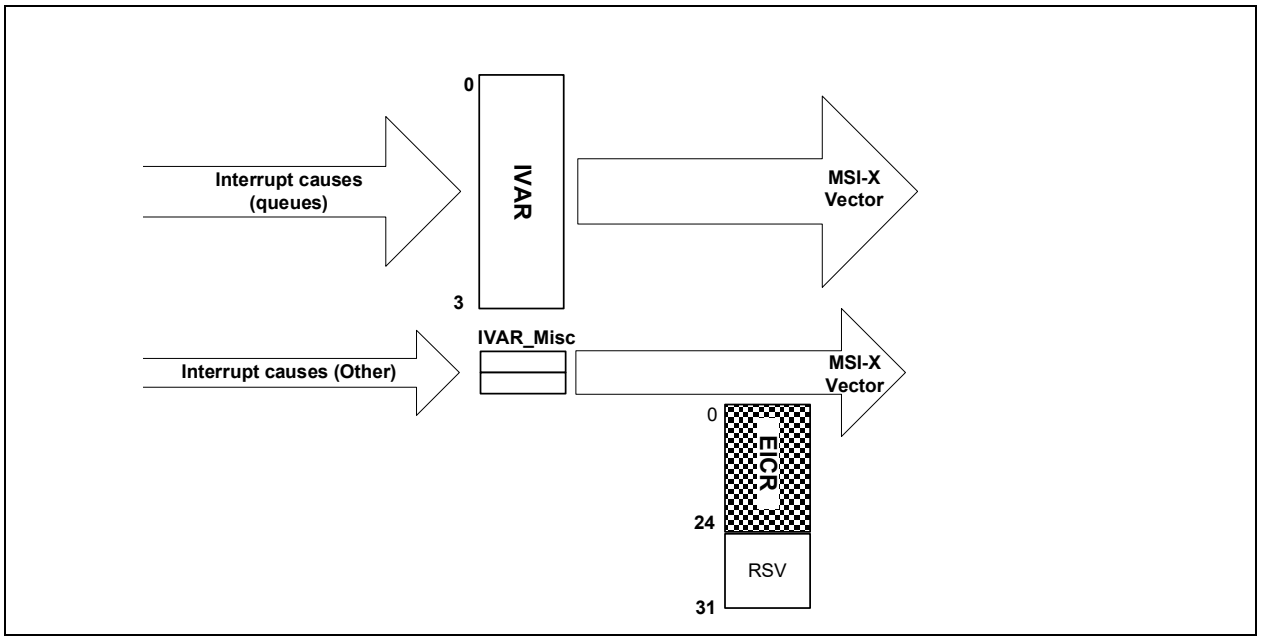
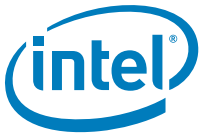


Figure 7-10. Cause Mapping in MSI-X Mode

Table 7-51 lists which interrupt cause is represented by each entry in the MSI-X Allocation registers. The software has access to 10 mapping entries to map each cause to one of the 5 MSI-x vectors.

Table 7-51. Cause Allocation in the IVAR Registers

Interrupt	Entry	Description
Rx_i	INT_Alloc[2*i] (i = 0..3)	Receive queues i - Associates an interrupt occurring in the RX queue i with a corresponding entry in the MSI-X Allocation registers.
Tx_i	INT_Alloc[2*i+1] (i = 0..3)	Transmit queues i- Associates an interrupt occurring in the TX queues i with a corresponding entry in the MSI-X Allocation registers.
TCP timer	INT_Alloc[8]	TCP Timer - Associates an interrupt issued by the TCP timer with a corresponding entry in the MSI-X Allocation registers.
Other cause	INT_Alloc[9]	Other causes - Associates an interrupt issued by the other causes with a corresponding entry in the MSI-X Allocation registers.



7.3.3 Legacy Interrupt Registers

The interrupt logic consists of the registers listed in Table 7-52 and Table 7-53, plus the registers associated with MSI/MSI-X signaling. Table 7-52 lists the use of the registers in legacy mode and Table 7-52 lists the use of the registers when using the extended interrupts functionality

Table 7-52. Interrupt Registers - Legacy Mode

Register	Acronym	Function
Interrupt Cause	ICR	Records interrupt conditions.
Interrupt Cause Set	ICS	Allows software to set bits in the ICR.
Interrupt Mask Set/Read	IMS	Sets or reads bits in the interrupt mask.
Interrupt Mask Clear	IMC	Clears bits in the interrupt mask.
Interrupt Acknowledge Auto-mask	IAM	Under some conditions, the content of this register is copied to the mask register following read or write of ICR.

Table 7-53. Interrupt Registers - Extended Mode

Register	Acronym	Function
Extended Interrupt Cause	EICR	Records interrupt causes from receive and transmit queues. An interrupt is signaled when unmasked bits in this register are set.
Extended Interrupt Cause Set	EICS	Allows software to set bits in the Interrupt Cause register.
Extended Interrupt Mask Set/ Read	EIMS	Sets or read bits in the interrupt mask.
Extended Interrupt Mask Clear	EIMC	Clears bits in the interrupt mask.
Extended Interrupt Auto Clear	EIAC	Allows bits in the EICR to be cleared automatically following an MSI-X interrupt without a read or write of the EICR.
Extended Interrupt Acknowledge Auto-mask	EIAM	This register is used to decide which masks are cleared in the extended mask register following read or write of EICR or which masks are set following a write to EICS. In MSI-X mode, this register also controls which bits in EIMC are cleared automatically following an MSI-X interrupt.
Interrupt Cause	ICR	Records interrupt conditions for special conditions - a single interrupt from all the conditions of ICR is reflected in the “other” field of the EICR.
Interrupt Cause Set	ICS	Allows software to set bits in the ICR.
Interrupt Mask Set/Read	IMS	Sets or reads bits in the other interrupt mask.
Interrupt Mask Clear	IMC	Clears bits in the Other interrupt mask.
Interrupt Acknowledge Auto-mask	IAM	Under some conditions, the content of this register is copied to the mask register following read or write of ICR.
General Purpose Interrupt Enable	GPIE	Controls different behaviors of the interrupt mechanism.

7.3.3.1 Interrupt Cause Register (ICR)

7.3.3.1.1 Legacy Mode

In Legacy mode, ICR is used as the sole interrupt cause register. Upon reception of an interrupt, the interrupt handling routine can read this register in order to find out what are the causes of this interrupt.

7.3.3.1.2 Advanced Mode

In advanced mode, this register captures the interrupt causes not directly captured by the EICR. These are infrequent management interrupts and error conditions.

Note that when EICR is used in advanced mode, the Rx /Tx related bits in ICR should be masked.

ICR bits are cleared on register read. If GPIE.NSICR = 0b, then the clear on read occurs only if no bit is set in the *IMS* register or at least one bit is set in the IMS register and there is a true interrupt as reflected in the *ICR.INTA* bit.

7.3.3.2 Interrupt Cause Set Register (ICS)

This register allows software to set bits in the ICR register. Writing a 1b in an *ICS* bit causes the corresponding bit in the ICR register to be set. Used usually to re-arm interrupts the software device driver didn't have time to handle in the current interrupt routine.

7.3.3.3 Interrupt Mask Set/Read Register (*IMS*)

An interrupt is enabled if its corresponding mask bit in this register is set to 1b, and disabled if its corresponding mask bit is set to 0b. A PCIe interrupt is generated whenever one of the bits in this register is set, and the corresponding interrupt condition occurs. The occurrence of an interrupt condition is reflected by having a bit set in the Interrupt Cause Register (ICR).

Reading this register returns which bits have an interrupt mask set.

A particular interrupt might be enabled by writing a 1b to the corresponding mask bit in this register. Any bits written with a 0b are unchanged. Thus, if software desires to disable a particular interrupt condition that had been previously enabled, it must write to the Interrupt Mask Clear (IMC) Register, rather than writing a 0b to a bit in this register.

7.3.3.4 Interrupt Mask Clear Register (IMC)

Software blocks interrupts by clearing the corresponding mask bit. This is accomplished by writing a 1b to the corresponding bit in this register. Bits written with 0b are unchanged (their mask status does not change).

7.3.3.5 Interrupt Acknowledge Auto-mask register (IAM)

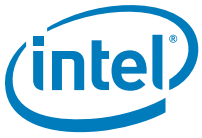
An ICR read or write has the side effect of writing the contents of this register to the IMC register to auto-mask additional interrupts from the *ICR* bits in the locations where the *IAM* bits are set. If GPIE.NSICR = 0b, then the copy of this register to the IMC register occurs only if at least one bit is set in the IMS register and there is a true interrupt as reflected in the *ICR.INTA* bit.

7.3.3.6 Extended Interrupt Cause Registers (EICR)

7.3.3.6.1 MSI/INT-A Mode (GPIE.Multiple_MSIX = 0b)

This register records the interrupts causes, to provide Software with information on the interrupt source.

The interrupt causes include:



1. The Receive and Transmit queues — Each queue (either Tx or Rx) can be mapped to one of the 4 interrupt causes bits (RxTxQ) available in this register according to the mapping in the *IVAR* registers
2. Indication for the TCP timer interrupt.
3. Legacy and other indications — When any interrupt in the Interrupt Cause register is active.

Writing a 1b clears the corresponding bit in this register. Reading this register auto-clears all bits.

7.3.3.6.2 MSI-X Mode (GPIE.Multiple_MSIX = 1b)

This register records the interrupt vectors currently emitted. In this mode only the first 5 bits are valid.

For all the subsequent registers, in MSI-X mode, each bit controls the behavior of one vector.

Bits in this register can be configured to auto-clear when the MSI-X interrupt message is sent, in order to minimize driver overhead when using MSI-X interrupt signaling.

Writing a 1b clears the corresponding bit in this register. Reading this register does not clear any bits.

7.3.3.7 Extended Interrupt Cause Set Register (EICS)

This register enables the software device driver to set *EICR* bits. Writing a 1b in a *EICS* bit causes the corresponding bit in the EICR register to be set. Used usually to re-arm interrupts that the software didn't have time to handle in the current interrupt routine.

7.3.3.8 Extended Interrupt Mask Set and Read Register (EIMS) & Extended Interrupt Mask Clear Register (EIMC)

Interrupts appear on PCIe only if the interrupt cause bit is a one and the corresponding interrupt mask bit is a one. Software blocks assertion of an interrupt by clearing the corresponding bit in the mask register. The cause bit stores the interrupt event regardless of the state of the mask bit. Different Clear (EIMC) and set (EIMS) registers make this register more “thread safe” by avoiding a read-modify-write operation on the mask register. The mask bit is set for each bit written as a one in the set register (EIMS) and cleared for each bit written as a one in the clear register (EIMC). Reading the set register (EIMS) returns the current mask register value.

7.3.3.9 Extended Interrupt Auto Clear Enable Register (EIAC)

Each bit in this register enables clearing of the corresponding bit in EICR following interrupt generation. When a bit is set, the corresponding bit in the EICR register is automatically cleared following an interrupt. This feature should only be used in MSI-X mode.

When used in conjunction with MSI-X interrupt vector, this feature allows interrupt cause recognition, and selective interrupt cause, without requiring software to read or write the EICR register; therefore, the penalty related to a PCIe read or write transaction is avoided.

See Section 7.3.4 for additional information on the interrupt cause reset process.

7.3.3.10 Extended Interrupt Auto Mask Enable Register (EIAM)

Each bit set in this register enables clearing of the corresponding bit in the extended mask register following read or write-to-clear to EICR. It also enables setting of the corresponding bit in the extended mask register following a write-to-set to EICS.

This mode is provided in case MSI-X is not used, and therefore auto-clear through EIAC register is not available.

In MSI-X mode, the driver software might set the bits of this register to select mask bits that must be reset during interrupt processing. In this mode, each bit in this register enables clearing of the corresponding bit in EIMC following interrupt generation.

7.3.3.11 GPIE Register

There are a few bits in the GPIE register that define the behavior of the interrupt mechanism. The setting of these bits is different in each mode of operation. Table 7-54 lists the recommended setting of these bits in the different modes:

Table 7-54. Settings for Different Interrupt Modes

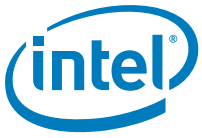
Field	Bit(s)	Initial Value	Description	INT-x/ MSI + Legacy	INT-x/ MSI + Extend	MSI-X Multi Vector	MSI-X Single Vector
NSICR	0	0b	Non Selective Interrupt clear on read: When set, every read of the ICR register clears the ICR register. When this bit is cleared, an ICR register read causes the ICR register to be cleared only if an actual interrupt was asserted or IMS = 0x0.	0b[1]	1b	1b	1b
Multiple_ MSIX	4	0b	Multiple_MSI-X - multiple vectors: 0b = non-MSI-X or MSI-X with 1 vector *IVAR* maps Rx/Tx causes to 4 *EICR* bits, but MSIX[0] is asserted for all. 1b = MSIX mode, IVAR maps Rx/Tx causes to 5 *EICR* bits. When set, the EICR register is not clear on read.	0b	0b	1b	0b
EIAME	30	0b	EIAME: When set, upon firing of an MSI-X message, mask bits set in EIAM associated with this message are cleared. Otherwise, EIAM is used only upon read or write of EICR/EICS registers.	0b	0b	1b	1b
PBA_ support	31	0b	PBA support: When set, setting one of the extended interrupts masks via EIMS causes the *PBA* bit of the associated MSI-X vector to be cleared. Otherwise, Foxville behaves in a way that supports legacy INT-x interrupts. Should be cleared when working in INT-x or MSI mode and set in MSI-X mode.	0b	0b	1b	1b

1. In systems where interrupt sharing is not expected, the *NSICR* bit can be set by legacy drivers also.

As this register affects the way the hardware interprets write operations to other interrupt control registers, it should be set to the correct mode before accessing other interrupt control registers.

7.3.4 Clearing Interrupt Causes

Foxville has three methods available to clear EICR bits: Auto-clear, clear-on-write, and clear-on-read. *ICR* bits might only be cleared with clear-on-write or clear-on-read.



7.3.4.1 Auto-Clear

In systems that support MSI-X, the interrupt vector allows the interrupt service routine to know the interrupt cause without reading the EICR. With interrupt moderation active, software load from spurious interrupts is minimized. In this case, the software overhead of a I/O read or write can be avoided by setting appropriate EICR bits to auto-clear mode by setting the corresponding bits in the Extended Interrupt Auto-clear Enable Register (EIAC).

When auto-clear is enabled for an interrupt cause, the *EICR* bit is set when a cause event mapped to this vector occurs. When the EITR Counter reaches zero, the MSI-X message is sent on PCIe. Then the *EICR* bit is cleared and enabled to be set by a new cause event. The vector in the MSI-X message signals software the cause of the interrupt to be serviced.

It is possible that in the time after the *EICR* bit is cleared and the interrupt service routine services the cause, for example checking the transmit and receive queues, that another cause event occurs that is then serviced by this ISR call, yet the *EICR* bit remains set. This results in a "spurious interrupt". Software can detect this case, for example if there are no entries that require service in the transmit and receive queues, and exit knowing that the interrupt has been automatically cleared. The use of interrupt moderations through the *EITR* register limits the extra software overhead that can be caused by these spurious interrupts.

7.3.4.2 Write to Clear

In the case where the driver wishes to configure itself in MSI-X mode to not use the "auto-clear" feature, it might clear the EICR bits by writing to the EICR register. Any bits written with a 1b is cleared. Any bits written with a 0b remain unchanged.

7.3.4.3 Read to Clear

The EICR and ICR registers are cleared on a read.

Note: The driver should never do a read-to-clear of the EICR when in MSI-X mode, since this might clear interrupt cause events which are processed by a different interrupt handler (assuming multiple vectors).

7.3.5 Interrupt Moderation

An interrupt is generated upon receiving of incoming packets, as throttled by the EITR registers (see Section 8.8.12). There is an *EITR* register per MSI-X vector.

In MSI-X mode, each active bit in EICR can trigger the interrupt vector it is allocated to. Following the allocation, the EITR corresponding to the MSI-X vector is tied to one or more bits in EICR.

When multi vector MSI-X is not activated, the interrupt moderation is controlled by register EITR[0].

Software can use EITR to limit the rate of delivery of interrupts to the host CPU. This register provides a guaranteed inter-interrupt delay between interrupts asserted by the network controller, regardless of network traffic conditions.

The following formula converts the inter-interrupt interval value to the common 'interrupts/sec.' performance metric:

$$\text{interrupts/sec} = (1 * 10^{-6}\text{sec} \times \text{interval})^{-1}$$

Note: In Foxville the interval granularity is 1 μs so some of the LSB bits of the interval are used for the low latency interrupt moderation.

For example, if the interval is programmed to 125d, the network controller guarantees the CPU is not interrupted by the network controller for at least 125 μs from the last interrupt. In this case, the maximum observable interrupt rate from the adapter should not exceed 8000 interrupts/sec.

Inversely, inter-interrupt interval value can be calculated as:

$$\text{inter-interrupt interval} = (1 * 10^{-6}\ \text{sec} \times \text{interrupt/sec})^{-1}$$

The optimal performance setting for this register is system and configuration specific.

The Extended Interrupt Throttle Register should default to zero upon initialization and reset. It loads in the value programmed by the software after software initializes the device.

When software wants to force an immediate interrupt, for example after setting a bit in the EICR with the Extended Interrupt Cause Set register, a value of 0 can be written to the Counter to generate an interrupt immediately. This write should include re-writing the *Interval* field with the desired constant, as it is used to reload the Counter immediately for the next throttling interval.

Foxville implements interrupt moderation to reduce the number of interrupts software processes. The moderation scheme is based on the EITR (Interrupt Throttle Register). Each time an interrupt event happens, the corresponding bit in the EICR is activated. However, an interrupt message is not sent out

on the PCIe interface until the *EITR* counter assigned to that *EICR* bit has counted down to zero. As soon as the interrupt is issued, the *EITR* counter is reloaded with its initial value and the process repeats again. The interrupt flow should follow Figure 7-11.

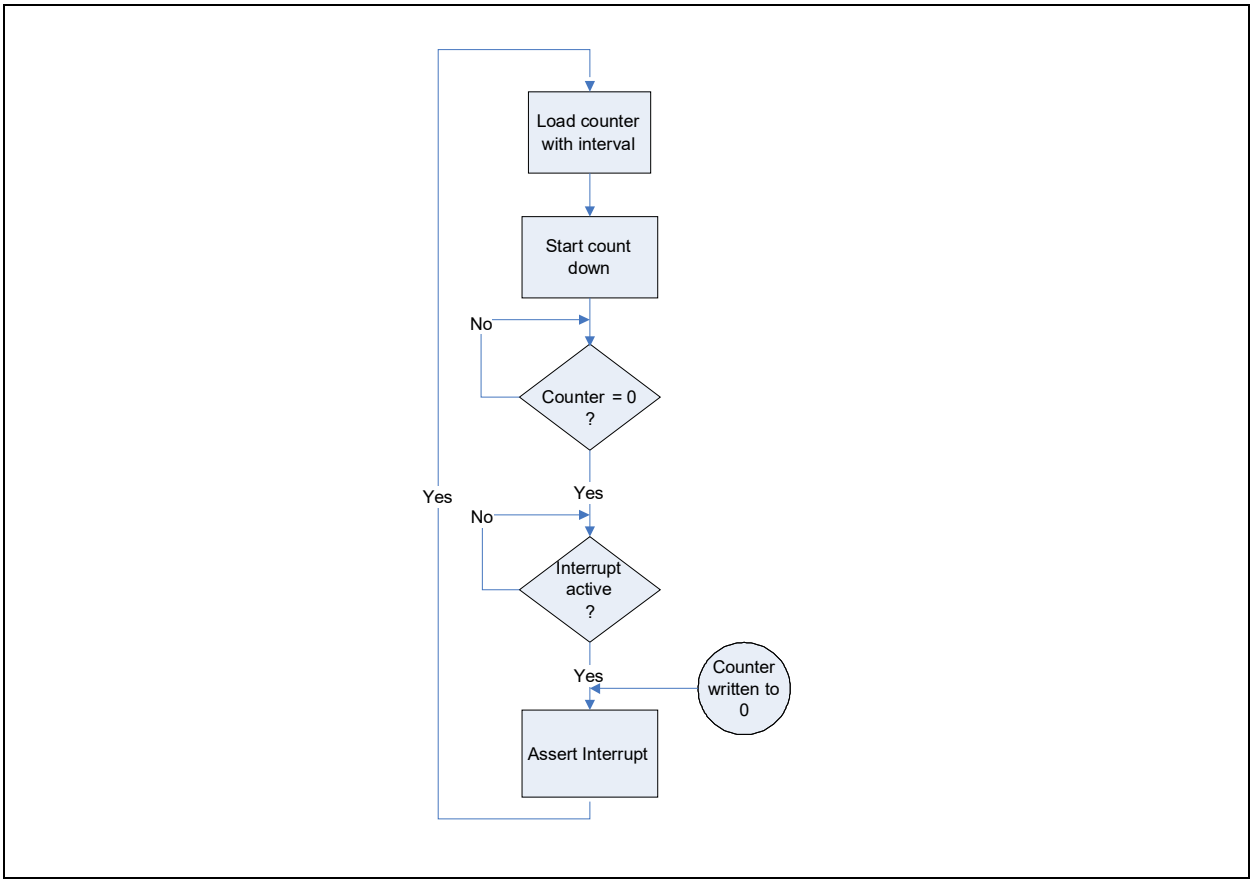


Figure 7-11. Interrupt Throttle Flow Diagram

EITR is designed to guarantee the total number of interrupts per second so for cases where Foxville is connected to a network with low traffic load, if the *EITR* counter counted down to zero and no interrupt event has happened, then the *EITR* counter is not re-armed but stays at zero. Thus, the next interrupt event triggers an interrupt immediately. That scenario is illustrated as Case B that follows.

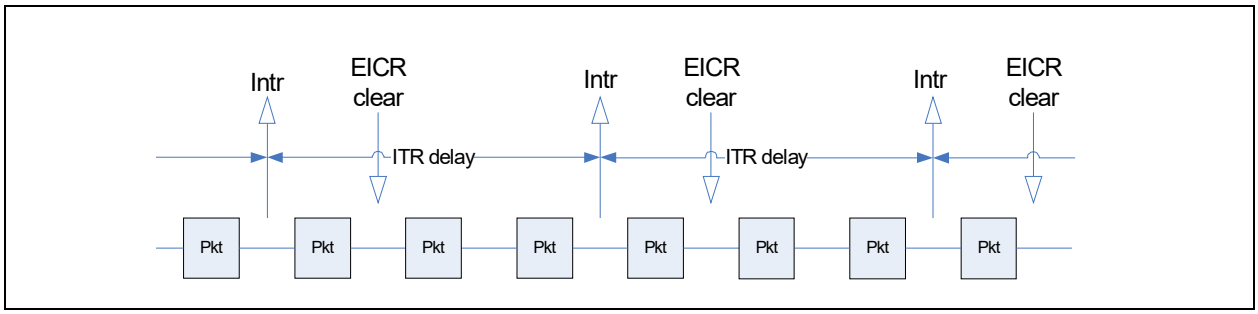


Figure 7-12. Case A: Heavy Load, Interrupts Moderated

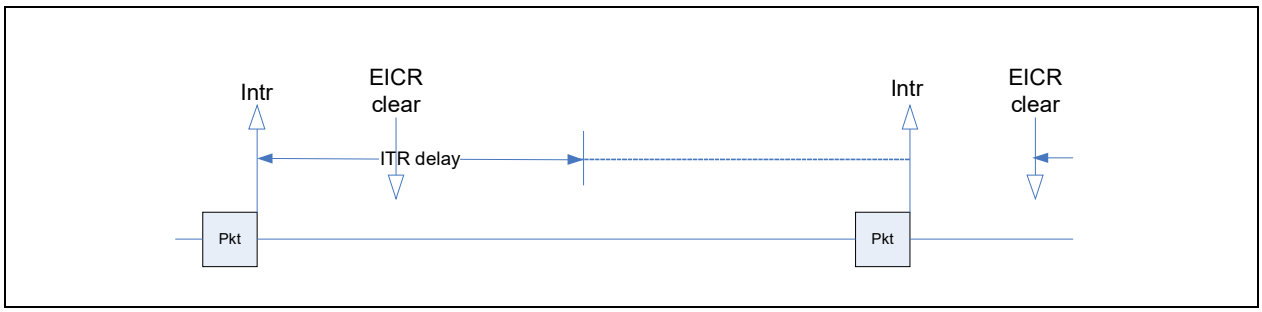


Figure 7-13. Light load, Interrupts Immediately on Packet Receive

7.3.6 Rate Controlled Low Latency Interrupts (LLI)

There are some types of network traffic for which latency is a critical issue. For these types of traffic, interrupt moderation hurts performance by increasing latency between the time a packet is received by hardware and the time it is handled to the host operating system. This traffic can be identified by the value, in conjunction with Control Bits and specific size. In addition packets with specific Ethernet types, TCP flag or specific VLAN priority might generate an immediate interrupt.

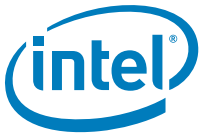
Low latency interrupts shares the filters used by the queueing mechanism described in Section 7.1.1. Each of these filters, in addition to the queueing action might also indicate matching packets might generate immediate interrupt.

If a received packet matches one of these filters, hardware should interrupt immediately, overriding the interrupt moderation by the *EITR* counter.

Each time a Low Latency Interrupt is fired, the *EITR* interval is loaded and down-counting starts again.

The logic of the low latency interrupt mechanism is as follows:

- There are 8 filters. The content of each filter is described in Section 7.1.2.4. The immediate interrupt action of each filter can be enabled or disabled. If one of the filters detects an adequate packet, an immediate interrupt is issued.



- There are 8 flex filters. The content of each filter is described in Section 7.1.2.5. The immediate interrupt action of each filter can be enabled or disabled. If one of the filters detects an adequate packet, an immediate interrupt is issued.
- When VLAN priority filtering is enabled, VLAN packets must trigger an immediate interrupt when the VLAN Priority is equal to or above the VLAN priority threshold. This is regardless of the status of the of the 2-tuple or Flex filters.
- The SYN packets filter defined in Section 7.1.2.6 and the ethernet type filters defined in section Section 7.1.2.3 might also be used to indicate low latency interrupt conditions.

Note: Immediate interrupts are available only when using advanced receive descriptors and not for legacy descriptors.

Note: Packets that are dropped or have errors do not cause a Low Latency Interrupt.

7.3.6.1 Rate Control Mechanism

In a network with lots of latency sensitive traffics the Low Latency Interrupt can eliminate the Interrupt throttling capability by flooding the Host with too many interrupts (more than the Host can handle).

In order to mitigate the above, Foxville supports a credit base mechanism to control the rate of the Low Latency Interrupts.

Rules:

- The default value of each counter is 0b (no moderation). This also preserves backward compatibility.
- The counter increments at a configurable rate, and saturates at the maximum value (31d).
  - The configurable rate granularity is 4 μs (250K interrupt/sec. down to 250K/32 ~ 8K interrupts per sec.).
- A LLI might be issued as long as the counter value is strictly positive (> zero).
  - The credit counter allows bursts of low latency interrupts but the interrupt average are not more than the configured rate.
- Each time a Low Latency Interrupt is fired the credit counter decrements by one.
- Once the counter reaches zero, a low latency interrupt cannot be fired
  - Must wait for the next ITR expired or for the next incrementing of this counter (if the EITR expired happened first the counter does not decrement).

The *EITR* and *GPIE* registers manage rate control of *LLI*:

- The *LL Interval* field in the GPIE register controls the rate of credits
- The 5-bit *LL Counter* field in the *EITR* register contains the credits

7.3.7 TCP Timer Interrupt

7.3.7.1 Introduction

The TCP Timer interrupt provides an accurate and efficient way for a periodic timer to be implemented using hardware. The driver would program a timeout value (usual value of 10 ms), and each time the timer expires, hardware sets a specific bit in the *EICR*. When an interrupt occurs (due to normal interrupt moderation schemes), software reads the *EICR* and discovers that it needs to process timer events during that DPC.

The timeout should be programmable by the driver, and the driver should be able to disable the timer interrupt if it is not needed.

7.3.7.2 Description

A stand-alone down-counter is implemented. An interrupt is issued each time the value of the counter is zero.

The software is responsible for setting initial value for the timer in the *TCPTIMER.Duration* field. Kick-starting is done by writing a 1b to the *TCPTIMER.KickStart* bit.

Following the kick-start, an internal counter is set to the value defined by the *TCPTIMER.Duration* field. Then during the count operation, the counter is decreased by one each millisecond. When the counter reaches zero, an interrupt is issued (see EICR register Section 8.8.1). The counter re-starts counting from its initial value if the *TCPTIMER.Loop* field is set.

7.3.8 Setting Interrupt Registers

In each mode, the registers controlling the interrupts should be set in a different way to assure the right behavior.

Table 7-55. Registers Settings for Different Interrupt Modes

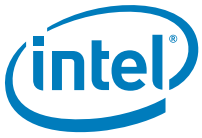
Field	Description	INT-x/MSI + Legacy	INT-x/ MSI + Extend	MSI-X Multi vector	MSI-X Single vector
IMS	Legacy Masks	Set[1]	Set[2]	Set[2]	Set[2]
IAM	Legacy Auto Mask Register	Might be set	0x0	0x0	0x0
EIMS	Extended Masks	Set Other Cause only.	Set[1]	Set[1]	Set[1]
EIAC	Extended Auto Clear register	0x0	0x0	At least one [3]	0x0
EIAM	Extended Auto Mask Register	0x0	Set[1]		Set[1]
EITR[0]	Interrupt Moderation register	Might be enabled	Might be enabled	Enable[4]	Enable
EITR[1...n]	Extended Interrupt Moderation register	Disable	Disable	Enable[4]	Disable
GPIE	Interrupts configuration	See Table 7-54 for details			

1. According to the requested causes
2. Only non traffic causes.
3. EIAC or EIAM or both should be set for each cause.
4. EITR must be enabled if Auto Mask is disabled. If Auto Mask is enabled, moderation might be disabled for the specific vector.

7.4 802.1Q VLAN Support

Foxville provides several specific mechanisms to support 802.1Q VLANs:

- Optional adding (for transmits) and stripping (for receives) of IEEE 802.1Q VLAN tags.
- Optional ability to filter packets belonging to certain 802.1Q VLANs.
- Double VLAN Support.



7.4.1 802.1Q VLAN Packet Format

The following diagram compares an untagged 802.3 Ethernet packet with an 802.1Q VLAN tagged packet:

Table 7-56. Comparing Packets

802.3 Packet	#Octets		802.1Q VLAN Packet	#Octets
DA	6		DA	6
SA	6		SA	6
Type/Length	2		802.1Q Tag	4
Data	46-1500		Type/Length	2
CRC	4		Data	46-1500
			CRC*	4

Note: The CRC for the 802.1Q tagged frame is re-computed, so that it covers the entire tagged frame including the 802.1Q tag header. Also, max frame size for an 802.1Q VLAN packet is 1522 octets as opposed to 1518 octets for a normal 802.3z Ethernet packet.

7.4.2 802.1Q Tagged Frames

For 802.1Q, the *Tag Header* field consists of four octets comprised of the Tag Protocol Identifier (TPID) and Tag Control Information (TCI); each taking 2 octets. The first 16 bits of the tag header makes up the TPID. It contains the "protocol type" which identifies the packet as a valid 802.1Q tagged packet.

The two octets making up the TCI contain three fields:

- User Priority (UP)
- Canonical Form Indicator (CFI). Should be 0b for transmits. For receives, the device has the capability to filter out packets that have this bit set. See the *CFIEN* and *CFI* bits in the *RCTL* described in Section 8.9.1.
- VLAN Identifier (VID)

The bit ordering is as follows:

Table 7-57. TCI Bit Ordering

Octet 1								Octet 2							
UP			CFI	VID											

7.4.3 Transmitting and Receiving 802.1Q Packets

7.4.3.1 Adding 802.1Q Tags on Transmits

Software might command Foxville to insert an 802.1Q VLAN tag on a per packet or per flow basis. If the *VLE* bit in the transmit descriptor is set to 1b, then Foxville inserts a VLAN tag into the packet that it transmits over the wire. 802.1Q tag insertion is done in different ways for legacy and advanced Tx descriptors:

- Legacy Transmit Descriptors:, The Tag Control Information (TCI) of the 802.1Q tag comes from the *VLAN* field (see Figure 7-10) of the descriptor. Refer to Table 7-28, for more information regarding hardware insertion of tags for transmits.
- Advanced Transmit Descriptor: The Tag Control Information (TCI) of the 802.1Q tag comes from the VLAN *Tag* field (see Table 7.2.2.2.1) of the advanced context descriptor. The *IDX* field of the advanced Tx descriptor should be set to the adequate context.

7.4.3.2 Stripping 802.1Q Tags on Receives

Software might instruct Foxville to strip 802.1Q VLAN tags from received packets. If VLAN stripping is enabled and the incoming packet is an 802.1Q VLAN packet (its *Ethernet Type* field matched the VET), then Foxville strips the 4 byte VLAN tag from the packet, and stores the TCI in the VLAN *Tag* field (see Figure 7-6 and See "Receive UDP Fragmentation Checksum) of the receive descriptor.

Foxville also sets the *VP* bit in the receive descriptor to indicate that the packet had a VLAN tag that was stripped. If the *CTRL.VME* bit is not set, the 802.1Q packets can still be received if they pass the receive filter, but the VLAN tag is not stripped and the *VP* bit is not set. Refer Figure 7-58 for more information regarding receive packet filtering.

4. VLAN stripping can be enabled By setting the *CTRL.VME* bit.

7.4.4 802.1Q VLAN Packet Filtering

VLAN filtering is enabled by setting the *RCTL.VFE* bit to 1b. If enabled, hardware compares the type field of the incoming packet to a 16-bit field in the VLAN Ether Type (VET) register. If the VLAN type field in the incoming packet matches the VET register, the packet is then compared against the VLAN Filter Table Array (VFTA[127:0]) for acceptance.

Foxville provides exact VLAN filtering for VLAN tags for host traffic and VLAN tags for manageability traffic.

7.4.4.1 Host VLAN Filtering:

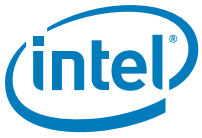
The *Virtual LAN ID* field indexes a 4096 bit vector. If the indexed bit in the vector is one; there is a Virtual LAN match. Software might set the entire bit vector to ones if the node does not implement 802.1Q filtering. The register description of the VLAN Filter Table Array is described in detail in Section 8.9.16.

In summary, the 4096-bit vector is comprised of 128, 32-bit registers. The *VLAN Identifier (VID)* field consists of 12 bits. The upper 7 bits of this field are decoded to determine the 32-bit register in the VLAN Filter Table Array to address and the lower 5 bits determine which of the 32 bits in the register to evaluate for matching.

7.4.4.2 Manageability VLAN Filtering:

The MC configures Foxville with eight different manageability VIDs via the Management VLAN TAG Value [7:0].

Two other bits in the Receive Control register (see Section 8.9.1), CFIEN and CFI, are also used in conjunction with 802.1Q VLAN filtering operations. CFIEN enables the comparison of the value of the *CFI* bit in the 802.1Q packet to the Receive Control register *CFI* bit as acceptance criteria for the packet.



Note: The *VFE* bit does not affect whether the VLAN tag is stripped. It only affects whether the VLAN packet passes the receive filter.

Table 7-58 lists reception actions per control bit settings.

Table 7-58. Packet Reception Decision Table

Is packet 802.1Q?	CTRL. VME	RCTL. VFE	Action
No	X[1]	X[1]	Normal packet reception
Yes	0b	0b	Receive a VLAN packet if it passes the standard MAC address filters (only). Leave the packet as received in the data buffer. *VP* bit in receive descriptor is cleared.
Yes	0b	1b	Receive a VLAN packet if it passes the standard filters and the VLAN filter table. Leave the packet as received in the data buffer (the VLAN tag would not be stripped). *VP* bit in receive descriptor is cleared.
Yes	1b	0b	Receive a VLAN packet if it passes the standard filters (only). Strip off the VLAN information (four bytes) from the incoming packet and store in the descriptor. Sets *VP* bit in receive descriptor.
Yes	1b	1b	Receive a VLAN packet if it passes the standard filters and the VLAN filter table. Strip off the VLAN information (four bytes) from the incoming packet and store in the descriptor. Sets *VP* bit in receive descriptor.

1. X - Don’t care

Note: A packet is defined as a VLAN/802.1Q packet if its type field matches the VET.

7.4.5 Double VLAN Support

Foxville supports a mode where most of the received and sent packet have at least one VLAN tag in addition to the regular tagging which might optionally be added. This mode is used for systems where the switches add an additional tag containing switching information.

Note: The only packets that might not have the additional VLAN are local packets that does not have any VLAN tag.

This mode is activated by setting *CTRL_EXT.EXT_VLAN* bit. The default value of this bit is set according to the EXT_VLAN (bit 1) in the *Initialization Control 3* NVM word.

The type of the VLAN tag used for the additional VLAN is defined in the *VET.VET_EXT* field.

7.4.5.1 Transmit Behavior with External VLAN

It is expected that the driver include the external VLAN header as part of the transmit data structure. Software might post the internal VLAN header as part of the transmit data structure or embedded in the transmit descriptor (see Section 7.2.2 for details). Foxville does not relate to the external VLAN header other than the capability of “skipping” it for parsing of inner fields.

Notes:

- If the *CTRL_EXT.EXT_VLAN* bit is set the VLAN header in a packet that carries a single VLAN header is treated as the external VLAN.
- If the *CTRL_EXT.EXT_VLAN* bit is set Foxville expects that any transmitted packet to have at least the external VLAN added by the software. For those packets where an external VLAN is not present, any offload that relates to inner fields to the EtherType might not be provided.

- If the regular VLAN is inserted from the descriptor (see Section 7.4.3.1), and the packet does not contain an external VLAN, the packet is dropped, and if configured, the queue from which the packet was sent is disabled.

7.4.5.2 Receive Behavior with External VLAN

When Foxville is working in this mode, it assumes that all packets received have at least one VLAN, including a packet received or sent on the manageability interface.

One exception to this rule are flow control PAUSE packets which are not expected to have any VLAN. Other packets might contain no VLAN, however a received packet that does not contain the first VLAN is forwarded to the host but filtering and offloads are not applied to this packet.

See Table 7-59 for the supported receive processing functions when the device is set to "Double VLAN" mode.

Stripping of VLAN is done on the second VLAN if it exists. All the filtering functions of Foxville ignore the first VLAN in this mode.

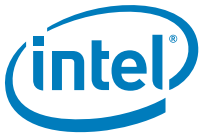
The presence of a first VLAN tag is indicated it in the *RDESC.STATUS.VEXT* bit.

Queue assignment of the Rx packets is not affected by the external VLAN header. It might depend on the internal VLAN, MAC address or any upper layer content as described in Section 7.1.1.

Table 7-59. Receive Processing in Double VLAN Mode

VLAN Headers	Status.VEXT	Status.VP	Packet Parsing	Rx Offload Functions
External and internal	1	1	+	+
Internal Only	Not supported			
V-Ext	1	0	+	+
None[1]	0	0	+ (flow control only)	-

1. A few examples for packets that might not carry any VLAN header might be: Flow control and Priority Flow Control; LACP; LLDP; GMRP; 802.1x packets



7.5 Time Sensitive Network Support - TSN

Foxville predecessors supported 1588 and scheduling at a basic level addressing mainly the audio video markets demands. Foxville is planned to address higher precision markets that require the following standards. The following subsections provide details on its support in Foxville:

- **1588** - Basic time-sync support (supported by Foxville predecessor as well). Details in the following subsections.
- **802.1AS-Rev** - Higher precision time synchronization with multiple (dual) clock masters. Details in the following subsections.
- **802.1Qav** - Credit Base scheduling is supported by up to two selected transmit queues as described in Section 7.5.2. This scheme is supported by Foxville predecessor as well.
- **802.1Qbv + Scheduling hints** - Time Aware Shaper, Adding time window associated with each transmit queue. Traffic from each queue can be transmitted only inside its own time window. On top of it, a unique launch time can be defined within the time window per each packet. The time aware shaper implementation is described in Section 7.5.2.9.
- **802.3Qbu, 802.3br** - Interspersing Express Traffic. A mechanism by which express traffic can preempt non-express traffic. Transmit queues are defined as carrying Express or non-express traffic. The implementation is described in Section 7.5.3.1.

Note: TSN is supported only in FDX (full duplex) network

7.5.1 Time SYNC (IEEE1588 and IEEE 802.1AS-Rev)

IEEE 1588 is a standard that provides a mechanism to synchronize clocks between remote nodes over symmetric FDX network links. One of the nodes plays the role of a clock master and the other nodes synchronize their clock to that master. The 802.1AS-Rev talks about unique profiling of the 1588 as well as supporting multiple clock masters. Foxville supports up to 2 x simultaneous clock masters. The following subsections describe the logic used for the 1588 clock implementation.

Important Notation Note: The 1588 protocol is also called PTP (Precision Time Protocol). However, in this document, we use the terms 1588 and PTP differently:

1588 timer is used to indicate the hardware timer implemented in the device.
PTP term is used to indicate a possible use case of the implemented 1588 timer.

Reader Note: Background information on the 1588 functionality and 802.1AS-Rev profiling is available in the matching standards.

7.5.1.1 PTP Time Synchronization Flow

The synchronization flow is achieved by 2 steps: matching the clock speed and then adjust the timer.

Matching the Speed of the Slave Clock to the Master

The master sends periodic SYNC packets as illustrated in Figure 7-14. Both the master and the slave sample the SYNC packet time: The master samples the packet transmission time and the slave samples the packet reception time. The master delivers the packet transmission time to the slave in one of two methods called: "**1 step**" or "**2 step**" (described in Section 7.5.1.2). Once the slave collects the master time as well as its own time it tunes its local 1588 clock rate to the ones of the master: adjusting the delta $T2_{n+1}$ minus $T2_n$ (shown in Figure 7-14) to be equals to $T1_{n+1}$ minus $T1_n$. The hardware mechanism supporting this process is described in Section 3.1.4.

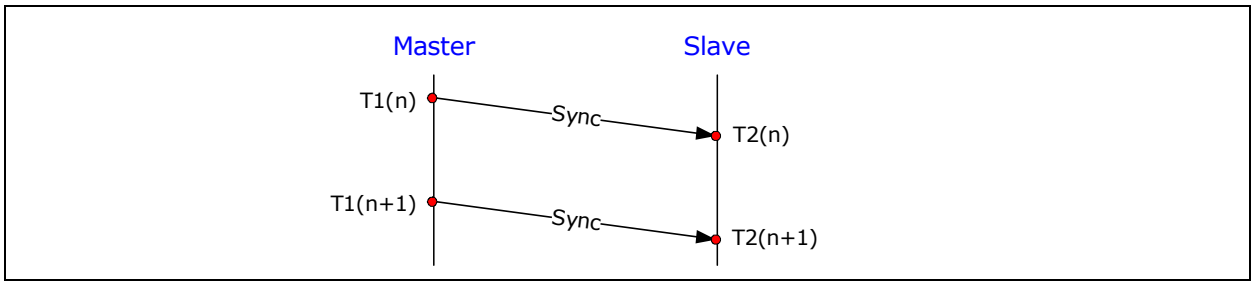


Figure 7-14. Matching the clock speed flow

Adjusting the Slave Timer to the Master

This phase starts after the clock speed of the slave is considered as identical to the clock speed of the master. The master continues to generate the periodic SYNC packets to the slave. Then the slave responds to each SYNC packet by a Delay_Req packet (illustrated in Figure 7-15). Both the slave and the master sample the time of these packets and these times are delivered to the slave. The slave computes the offset between its own 1588 timer and the master based on the following equation:

Time Offset between the master and the slave = [(T2-T1) - (T4-T3)] / 2

- **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)

A basic assumption behind the above equation is that the latency of the traffic from the Master to the Slave is identical to the inverse traffic latency from the Slave to the Master. This assumption is called "symmetric link". In some cases, the link does not behaves as a "symmetric link" (mainly the latency of the LAN controllers). In such a case, appropriate calibration should be made.

The slave can calibrate its own timer to the master in one of two methods:

1. Fine calibration of the clock speed aiming for calibrated timer down the line.
2. Adjustment of the 1588 timer. The adjustment could be the whole offset value between the master and the slave or a kind of a low pass filtered version of it.

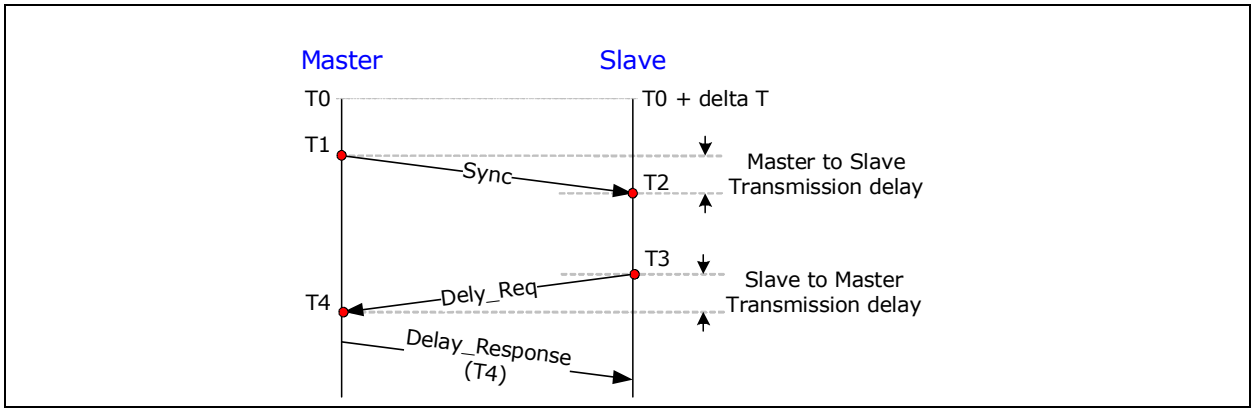


Figure 7-15. Sync Flow and Offset Calculation

7.5.1.2 Time-Stamp Delivery from the Master to the Slave

The slave synchronizes its timer to the master based on the packet time-stamps sampled by the slave as well as the time-stamps sampled by the master. In order to do so, the slave must have these time-stamp values. The master sends the time-stamp of the Delay_Req packet reception on a regular data packet called "Delay_Response". The time-stamp of the SYNC packet can be sent to the slave in one of two methods (illustrated in Figure 7-16):

3. Using the 2 step method, the master sends the time-stamp of the SYNC packet in a "Follow_Up" data packet.
4. Using the 1 step method, the master sends the time-stamp of the SYNC packet in the SYNC packet itself. This method requires from the hardware to append the time-stamp of the packet to the same packet itself during its transmission.

Foxville supports both methods while the "1 step" method is supported only when the SYNC packets are sent over L2 or over UDP without UDP checksum.

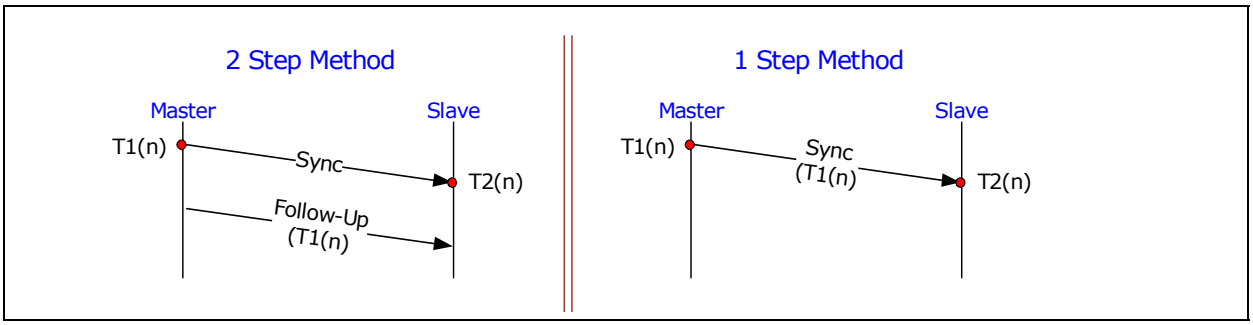


Figure 7-16. 1 Step and 2 Step flows

7.5.1.3 1588 Timer - Logic and Its programming

The following subsections describe the 1588 hardware elements and its related programming. The whole 1588 logic is enabled only when the Disable systime bit in the TSAUXC register is cleared.

7.5.1.3.1 1588 Timers Logic

Foxville contains 4 1588 timers that can operate simultaneously. There are several applications that can use multiple timers. Two examples for possible use cases for the multiple timers:

use case 1: One timer is synchronized to an active PTP master and the other one is synchronized to a a backup PTP master for fast transition between the masters. In this case it is expected that when both masters are functional, the two 1588 timers are almost identical.

use case 2: One timer is used to synchronize with one PTP master timer and the other one could be used as a global timer. In this case the timers could be completely non-synchronized to one another.

use case 3: One of the timers could be used as non-synchronized monotonic timer.

The Timer's Counters - SYSTIM: The 1588 timers are consist of a 96 bit counter each. Implemented by 4 sets of 3 counters: *SYSTIMR*, *SYSTIML* and *SYSTIMH*. The *SYSTIMR* register is designed to hold the sub ns fraction, the *SYSTIML* register holds the ns fraction and the *SYSTIMH* register holds the second fraction of the time. The upper two bits of the *SYSTIML* register are always zero while the max value of this register is 999,999,999 dec. On top of it, there is an additional 16 bit register (managed by

software) that holds the upper 2 bytes of the standard 1588 time. When synchronized to the global time, the *SYSTIM* registers holds the absolute time relative to January 1st 1970 00:00:00 International Atomic Time (TAI).

Initial Setting: Software can set an initial time value to the SYSTIM registers by direct write access. It makes sense to set only the *SYSTIMH register (the sec units of the timer). Setting the sub sec units can be made by the "Time adjust" procedure described below.*

***Time adjust**: Adjustment of the SYSTIM timer by a relative value to the timer value is done during nominal operation by the TIMADJ register as described* below.

Reading the SYSTIM: Read the *SYSTIML* register and then the SYSTIMH register. Reading the SYSTIML register, the upper 32bits are latched to a SYSTIMH shadow register for coherent context. Note that there are 4 SYSTIMH shadow registers supporting the 4 sets of SYSTIM registers.

The Timer's Step - TIMINCA: Each 1588 timer clock the SYSTIM is incremented by "some" value that defines the SYSTIM in nsec units. The increment value is composed of two parts:

- H_INC Hard coded value as shown in Table 7-60. The units of the H_INC are the same as the SYSTIML (which is nsec units).
- TIMINCA A programmable portion enabling to synchronize the rate of the timer to another timer.

The SYSTIM is incremented each clock according to the following equation:

SYSTIM = SYSTIM + H_INC + TIMINCA.Incvalue if TIMINCA.ISGN = 0 or . . .
SYSTIM = SYSTIM + H_INC - TIMINCA.Incvalue if TIMINCA.ISGN = 1

The units of H_INC are as SYSTIML and the units of Incvalue are as SYSTIMR

Table 7-60. H_INC value per device SKU

Foxville SKU -->	2.5G SKUs	
1588 timer Clock Frequency	312.5 MHz	
H_INC value	Sequence of 5 values for 5 consecutive clocks: 3.25 , 3.25 , 3.25 , 3.25 , 3	

The Timer's Adjust Parameter - TIMADJ: Software can adjust the 1588 timer by some value using the TIMADJ register. There are 4 TIMADJ registers supporting the 4 sets of SYSTIM registers. It can be programmed to some positive or negative adjustment value. Setting the TIMADJ register, the SYSTIM registers are updated once in the next 1588 clock by the regular Incvalue plus the value of the TIMADJ register. The TIMADJ register is defined in the same units that the *SYSTIML register is defined: using the* Incvalue recommended in the above Table 7-60, the TIMADJ register is defined in nsec units.

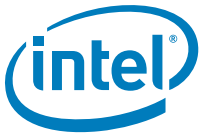
7.5.1.3.2 1588 Timers Logic Supporting 4 x 1588 Timers

Supporting 4 1588 timers, Foxville includes 4 sets of the 1588 timer registers. That includes the following registers:

- SYSTIMTM, SYSTIMH, SYSTIML, SYSTIMR, TIMINCA and TIMADJ.

On top of it, additional logic is associated to one of the 4 1588 timers. This include the following:

- The transmit queues are associated to one of the timers by TXQCTL.Report_TIMER_SEL. This setting affect the reported DMA completion of the packet data in the transmit descriptor.
- Transmit packets time stamp in the TXSTMP register is done by one of the timers selected by the 1588_Sel flag in the transmit data descriptor.



- Receive packets are sampled by both timers and reported in the "Receive Packet Timestamp in Buffer" of the receive data structure.
- The target time registers are associated to one of the timers by TRGTTIML.IO_TIMER_SEL.
- The sampling event registers are associated to one of the timers by *AUXSTMPL*.IO_TIMER_SEL.

7.5.1.3.3 Packet Time-Stamp Sampling by the Hardware

Foxville samples the packet transmission and reception time by a sync signal from the PHY logic providing as deterministic as possible delay from the sampling logic to the network pins. The latency and jitter between the sampled time-stamp and the beginning of the SFD symbol on the network pins is shown in the Table below:

Transmit Time-Stamp Sampling Latency

Link Speed -->	2.5G	1G	100M	10M
Latency from Time-Stamp and **transmit** SFD on the network pins				

Note: Values are still TBD

7.5.1.3.4 Receive Packet Sampling - Software Interface

Receive time-stamping is enabled per receive queue by the Timestamp flag in the SRRCTL register of the queue. The packet time-stamp is indicated by an active TSIP flag in the receive descriptor and the time-stamp value is posted to the SYSTIM fields in the data buffer. See Section 7.1.6 for complete description of the time-stamp reporting structure and additional required settings enabling it. The latency and jitter between the SFD symbol on the network pins and the sampled time-stamp is shown in the Table 7-61 below:

Table 7-61. Receive Time-Stamp Sampling Latency

Link Speed -->	2.5G	1G	100M	10M
Latency from received SFD on the network pins and Time-Stamp				

Note: Values are still TBD

7.5.1.3.5 Transmit Packet Sampling - Software Interface

For each packet that should be sampled, the PTP field in the Advanced Transmit Data Descriptor should be set as described in Section 7.2.2.3.3. Furthermore, sampling the time in the TXSTMP registers is enabled by the TSYNCTXCTL.EN flag (1 step functionality is enabled regardless of the TSYNCTXCTL.EN flag).

7.5.1.3.6 Synchronized SDP Output to the 1588 timers

The two target time registers *TRGTTIML/H0* and *TRGTTIML/H1* per timer enable generating a time triggered event on one of the SDP pins according to the setup defined in the *TSSDP* and *TSAUXC* registers (See Section 8.15.13 and Section 8.15.25). The target time registers are structured the same as the *SYSTIML/H* registers with additional IO_TIMER_SEL flag in the TRGTTIML registers associating the target time to one of the 1588 timers.

7.5.1.3.6.1 Shared Settings to all synchronized SDP output options

1. Select a specific SDP pin by setting the *TSSDP.TS_SDPx_EN* flag to 1b (while 'x' is 0, 1, 2 or 3).
2. Assign one of the 1588 timers by the IO_Timer_Sel field in the matched target time register.
3. Define the selected SDPx pin as output, by setting the appropriate *SDPx_IODIR* bit (while 'x' is 0, 1, 2, or 3) in the *CTRL* or *CTRL_EXT* registers.

7.5.1.3.6.2 Synchronized Level Changed on SDP Output

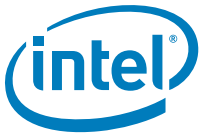
Required unique settings to generate a level change on one of the SDP pins when System Time (*SYSTIM*) reaches a pre-defined value, the driver should do the following:

1. Assign a target time register to the selected SDP by setting the *TSSDP.TS_SDPx_SEL* field to 00b or 01b if level change should occur based on *TRGTTIML/H0 or TRGTTIML/H1,* respectively.
2. Program the selected target time *TRGTTIML/Hx* (while 'x' is 0b or 1b) to the required event time.
3. Program the *TRGTTIML/Hx* to "Level Change" mode by setting the *TSAUXC.PLSG* bit to 0b and *TSAUXC.EN_TTx* bit to 1b (while 'x' is 0b or 1b).
4. When the *SYSTIML/H* registers cross the value of the *TRGTTIML/H* registers, the selected SDP changes its output level.

7.5.1.3.6.3 Synchronized Pulse on SDP Output

An output pulse can be generated by using one of the target time registers to define the beginning of the pulse and the other target time registers to define the pulse completion time. Required unique settings to generate a pulse on one of the SDP pins when System Time (*SYSTIM*) reaches a pre-defined value, the driver should do the following:

1. Select the target time register for the selected SDP that defines the beginning of the output pulse. It is done by setting the *TSSDP.TS_SDPx_SEL* field to 00b or 01b if level change should occur when *SYSTIML/H* equals *TRGTTIML/H0 or TRGTTIML/H1*, respectively.
2. Program the target time *TRGTTIML/Hx* (while 'x' is 0b or 1b) to the required event time. The registers indicated by the *TSSDP.TS_SDPx_SEL* define the leading edge of the pulse and the other ones define the trailing edge of the pulse.
3. Program the *TRGTTIML/Hx* defined by the *TSSDP.TS_SDPx_SEL* to "Start of Pulse" mode by setting the *TSAUXC.PLSG* bit to 1b and *TSAUXC.EN_TTx* bit to 1b (while 'x' is 0b or 1b). The other *TRGTTIML/Hx* register *should be set* to Level Change mode by setting the *TSAUXC.PLSG* bit to 0b and *TSAUXC.EN_TTx* bit to 1b (while 'x' is 0b or 1b).
4. When the *SYSTIML/H* registers cross the *value of the TRGTTIML/H* registers that define the beginning of the pulse, the selected SDP changes its level. Then, when the *SYSTIML/H* registers cross the other *TRGTTIML/H* registers (that define the trailing edge of the pulse), the selected SDP changes its level back.



7.5.1.3.6.4 Synchronized Output Clock on SDP Output

Foxville supports driving a programmable Clock on the SDP pins (up to two output clocks). The output clocks generated are synchronized to the global System time registers (*SYSTIM*). The Target Time registers (*TRGTTIML/H0* or *TRGTTIML/H1*) can be used for the clock output generation. To start an clock output on one of the SDP pins when System Time (*SYSTIM*) reaches a pre-defined value, the driver should do the following:

1. Select the target time register for a selected SDP, by setting the *TSSDP.TS_SDPx_SEL* field to 10b or 11b if output clock should occur based on *TRGTTIML/H0 or TRGTTIML/H1* respectively.
2. Program the matched *FREQOUT0/1* register to define clock half cycle time. Note that in the general case the maximum supported half cycle time of the synchronized output clock is 70 ms. A slower output clock can be generated by the Synchronized Level Change scheme described in Section 7.5.1.3.6.2. In this option, software should trigger the output level change time periodically for each clock transition. Slower half cycle time than 70msec can be programmed also as long as the output clock is synchronized to whole seconds as follow (useful specifically for generating a 1Hz clock):
   a. The clock should start at a programmable time (see bullet on TRGTTIML/Hx below)
   b. For larger *FREQOUT* values than 70msec, The starting time plus 'n' times the value of the programmed FREQOUT0/1 must be whole number of seconds (for 'specific' values of 'n')
   c. Permitted values for the FREQOUT0/1 register that can meet the above conditions are: 125,000,000 decimal, 250,000,000 decimal and 500,000,000 decimal (equals to 125msec, 250msec and 500msec respectively)
3. TRGTTIML/Hx should be set to the required start time of the clock.
4. Enabled the clock operation by setting the relevant *TSAUXC.EN_CLK0/1* bit to 1b.

The clock out initially drives a logic zero level on the selected SDP. When SYSTIM crossesthe TRGTTIM, the hardware begins an endless loop of the following two steps:

1. Revert the SDP output level.
2. Increment the used *TRGTTIML/Hx* by *FREQOUT* value for the next output clock phase.

7.5.1.3.7 Synchronized SDP Input to the 1588 timers

Upon a change in the input level of one of the SDP pins that was configured to detect Time stamp events using the TSSDP register, a time stamp of the system time is captured into one of the two auxiliary time stamp registers (*AUXSTMPL/H0* or *AUXSTMPL/H1*). The AUXSTMP registers are associated to one of the 1588 timers by the IO_TIMER_SEL flag in the AUXSTMPL registers. Software enables the timestamp of input event as follow:

1. Define the sampled SDP on AUX time register 'x' ('x' = 0b or 1b) by setting the *TSSDP.AUXx_SDP_SEL* field while setting the matched *TSSDP.AUXx_TS_SDP_EN* bit to 1b.
2. Set also the *TSAUXC.EN_TSx* bit ('x' = 0b or 1b) to 1b to enable "timestamping".
3. Define the 1588 timer 0 or timer 1 by the TS_SDPx_Timer flag of the selected SDP (x=0...3).

Following a transition on the selected SDP, the hardware does the following:

1. The SYSTIM registers (low and high) are latched to the selected AUXSTMP registers (low and high)
2. The selected AUTT0 or AUTT1 flags are set in the TSICR register. If the AUTT interrupt is enabled by the TSIM register and the 1588 interrupts are enabled by the Time_Sync flag in the ICR register then an interrupt is asserted as well.

After the hardware reports that an event time was latched, the software should read the latched time in the selected AUXSTMP registers. Software should read first the Low register and only then the High register. Reading the high register releases the registers to sample a new event.

7.5.1.3.8 Time SYNC Interrupts

Time Sync related interrupts can be generated by programming the *TSICR* and *TSIM* registers. The *TSICR* register logs the interrupt cause and the *TSIM* register enables masking specific *TSICR* bits. Detailed description of the Time Sync interrupt registers can be found in Section 8.16. Occurrence of a Time Sync interrupt sets the *ICR.Time_Sync* interrupt bit.

7.5.2 Transmit Scheduling (802.1Qav and 802.1Qbv)

This chapters talks about handling time sensitive streams as part of TSN functionality.

The 802.1Qav is part of the TSN specifications that include Timing and Synchronization for time specific applicationsand queuing enhancements for time sensitive streams. Foxville provides a mechanism by which the software can define a credit based scheduling or transmission time for packets within a transmit queue. Transmission time is defined for transmit queues that are set as scheduled queues. On top of it, Foxville supports the Time Aware Shaper scheme (802.1Qbv) enabling transmission only within permitted time windows. See Section 7.5.2.9 for description.

Note that when supporting Time Sensitive Streams, EEE should not be enabled (by the EEER register) due to potential distortions of the time sensitive traffic scheduling.

Note that transmit segmentation offload should not be enabled on queues that are subjected to scheduling (defined as TSN queues by the Queue_Mode flag in the TXQCTL register of the queue) and should not be enabled on PTP packets as well.

Note that the transmit scheduler operates on a 156.25 MHz clock which dictates its accuracy of 6.4nsec.

7.5.2.1 Foxville Transmit Buffer Modes

Foxville supports two transmit modes, legacy and TSN. The transmit mode is configured in TQAVCTRL.TRANSMIT_MODE register and must be set at SW initialization cycle. Foxville transmit mode cannot change during dynamic operation.

	Legacy	TSN
Packet Buffer	Single transmit packet buffer	Four transmit packet buffers
Queues	Four transmit queues All enabled queues are associated to the single transmit packet buffer	Four transmit queues Each enabled queue is associated with a dedicated transmit packet buffer
Data fetch arbitration	Round robin between the queues	Combination of time based and most empty packet buffer
Data transmit arbitration	None - single packet buffer - first in first out	Strict priority and time based arbitration for the TSN queues

Foxville Legacy transmit is defined in Section 7.2. The rest of this sub chapter defines Foxville transmit functionality when configured to operate at TSN mode.

7.5.2.2 Foxville Transmit Queue Types

Foxville supports two transmit queue types related to scheduling: Queues on which the launch time is defined (in the transmit descriptor) and queues with no launch time. These transmit setting options are configured per queue by the QUEUE_MODE flag in the TXQCTL register of the queues. Time based

scheduling is enabled only for those queues defined as LaunchT queues. Queues can be defined as LaunchT queues only when the transmit buffer is configured to TSN (by the global TQAVCTRL.TRANSMIT_MODE parameter).

7.5.2.3 Transmit Pipeline in TSN Mode

Priority and scheduling of the traffic is achieved by allocating 4 packet buffers, one per each of the host transmit queues, and additional packet buffer for the manageability transmit queue. Transmit arbitration of these packet buffers is based on the queue's priority and the programmed scheduling of the packets. The Figure 7-17 below illustrates the transmit arbitration scheme when the device is configured to handle time sensitive streams. It is then detailed in the following sections.

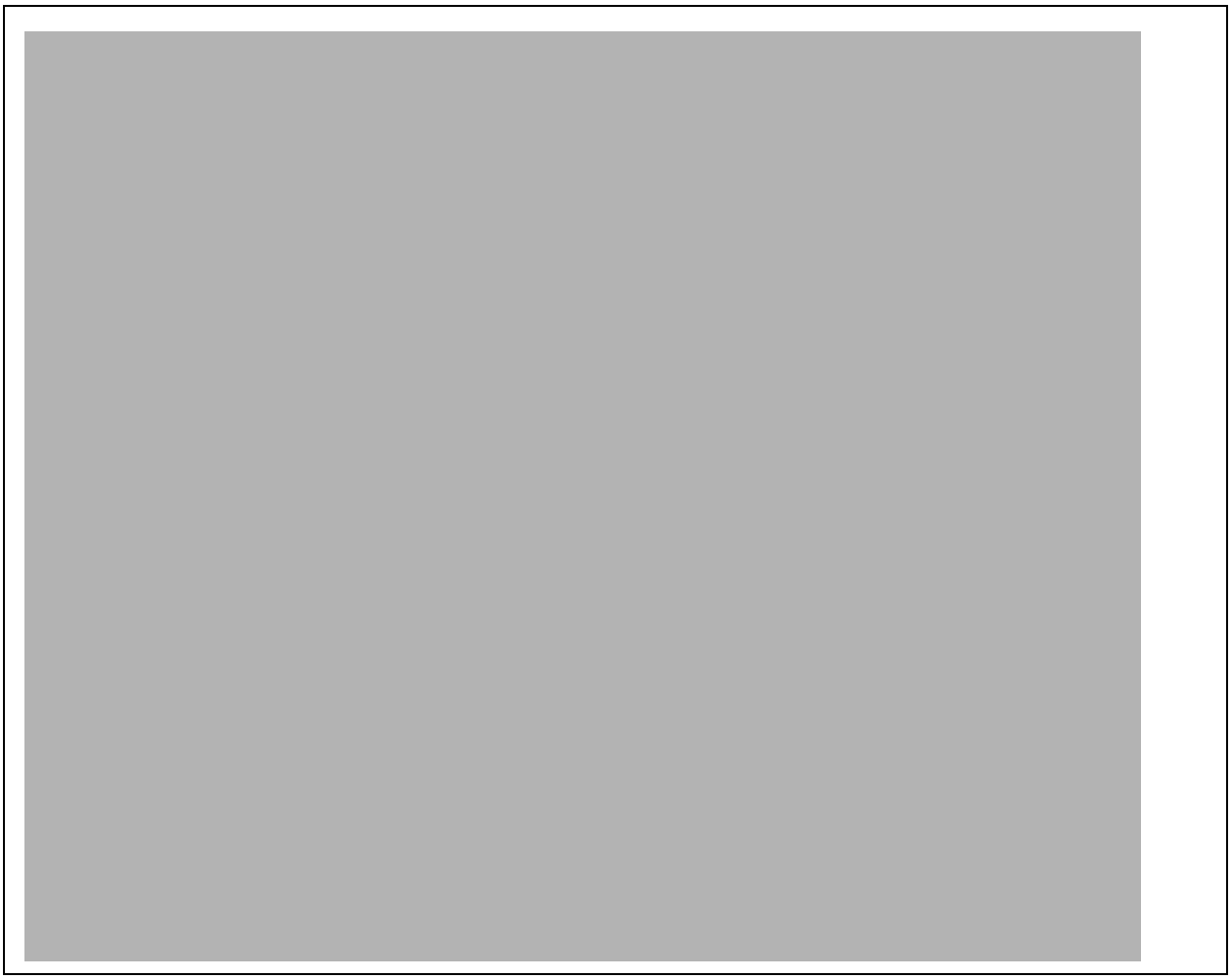


Figure 7-17. Transmit Architecture TSN Mode

7.5.2.4 Mapping User Priorities to Queues

While in TSN mode each transmit queue is assigned a priority level by the TxQ_Priority fields in the TxARB register. The transmit queues must be assigned unique TxQ_Priority levels. Each transmit queues can carry a single or multiple User Priorities. It is the software responsibility to map packets with specific user priority to the hardware queues.

7.5.2.5 Transmission Selection

Transmission selection is the process of selecting the next packet to transmit, transmission selection in Foxville includes three levels of arbitration: descriptor fetch, data fetch and data transmission. Transmission with scheduling options is enabled by the TQAVCTRL.TRANSMIT_MODE set to TSN.

Descriptor fetch - the transmit descriptor fetch mechanism while in TSN modes is the same as in legacy mode, the complete description of descriptor fetch is described in Section 7.2.2.5.

Data fetch - the data fetch mechanism while in TSN modes is the same as the legacy mode with some enhancements listed below:

- Arbitration between the queues is round robin or most empty packet buffer according to the TQAVCTRL.DATA_FETCH_ARB setting. Round robin option is between all queues with the same priority configured by the *TXDCTL[n].priority* while queues configured as high priority are always selected before queues with low priority (i.e., strict priority).
- On top of it, if the time based fetch is enabled by the TXQCTL.DATA_FETCH_TIM parameter, TSN queues are subjected to fetch time policy: Fetch is enabled only if the fetch time is expired as well. The fetch time is defined by the scheduling time minus the TXQCTL.FETCH_TIM_DELTA.

Data transmission - transmission arbitration flow is described below.

- In non TSN mode (TQAVCTRL.TRANSMIT_MODE = Legacy) there is only a single transmit packet buffer. Packets are transmitted in the same order they are fetched from host memory. Arbitration between Manageability traffic and host traffic is simple round robin.
- The following bullets relates to TSN mode ((TQAVCTRL.TRANSMIT_MODE = TSN)...
- Manageability traffic shares the transmission arbitration with one of the 4 transmit queues based on the “MNG_Arb” field in the TxARB register. The Manageability traffic has higher strict priority over the selected host queue. Once a packet is selected in this step, it inherit the selected host queue arbitration priority and all other TSN attributes in the following steps. Specifically it inherit the Qbv scheduling. The end of the Qbv window that is used for the Manageability traffic should be programmed to be 24 bytes shorter than the required end.It is software responsibility to reserved bandwidth for the MNG traffic. Failing to do so, the MNG traffic switches to an “emergency” mode and will transmit its packets bypassing any bandwidth, priority or Qbv windowing settings. In such a case, the host traffic might sense some scheduling distortions as well.
- There are 4 x priority levels: Each queue is mapped to a priority level by the “TxQ_Priority” fields in the TxARB register. Each queue must be assigned with a unique priority level. Queue selection for transmission is always strict. Priority level 0 always selected first (highest priority). For proper operation Express queue must be set with higher priority level than Non-Express queue.
- The queues are eligible for transmission if the following conditions are met:
  - The whole packet data is already fetched to the transmit packet buffer.
  - If credit based shaping is assigned for the queue (by the Qav_Sel field in the TXQCTL register) then the queue must have non-negative credit.
  - For TSN queues (Queue_Type = TSN), transmission time must be expired.
  - Transmission is enabled only inside the Qbv time window and obeying the Strict_End parameter of the queue (see description of the Qbv scheme in Section 7.5.2.9.2).

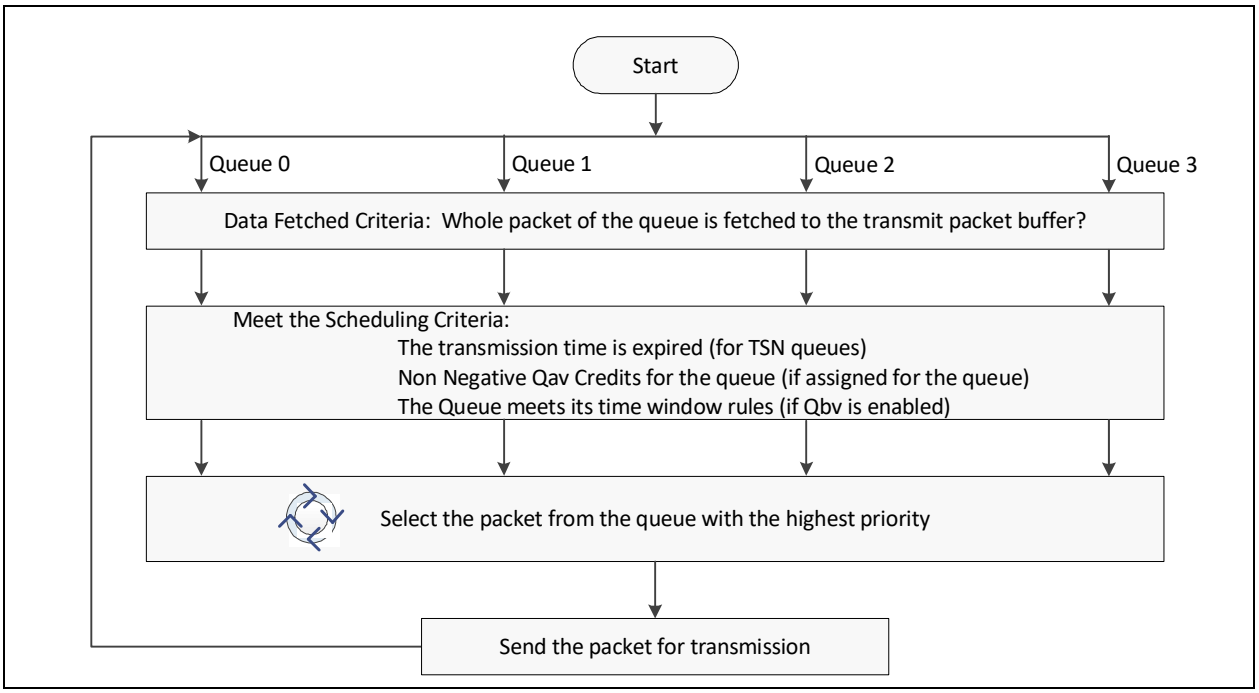


Figure 7-18. Data Transmission Arbitration Operation for TRANSMIT_MODE = TSN

7.5.2.6 Transmit Scheduling Latency

The latency between transmission scheduling (launch time) and the time the packet is transmitted to the network is listed in Table 7-61. Software can compensate it by setting the GTxOffset to the device latency that is given in Table 7-61.

Table 7-62. Packet Scheduling to its Transmission Latency

SKU	Link Speed	Latency between launch time and beginning of the SFD symbol on the MDIO pins		
		Min	Max	Comments
1G b/s	10 Mb/s			
	100 Mb/s			
	1 Gb/s			
2.5G b/s	10 Mb/s			
	100 Mb/s			
	1 Gb/s			
	2.5 Gb/s			

Note: Values are still TBD

7.5.2.7 Credit Based Shaping - 802.1Qav

Foxville supports 2 sets of credit based shaping logic. Each transmit queue can be associated with one of these 2 credit based shaping sets by the Qav_Sel parameter in the TXQCTL register. When credit based shaping is associated to a queue, transmission is enabled only if the credits of the queue are non-negative. The parameters below describes the scheme upon which credits are consumed and accumulated:

TQAVCC.IdleSlope - The Idle slop defines the amount of accumulated credits per transmission time of a byte while there is a packet to be transmitted within the Qbv window (if Qbv is enabled), and the queue is blocked from transmission. The IdleSlope settings is calculated using the following equation: **IdleSlope** = (Link Rate / 100Mb/s) * 0x7736 * BW * 0.2 / 2.5 , while BW is the target percentage link utilization

SendSlope - The Send slop defines the amount of credit consumption per transmitted byte and is calculated by the formula SendSlope = IdleSlope - LinkRate.

LinkRate - Constant at 0x7736.

TQAVHC.HiCredit - The HiCredit defines an upper limit for the accumulated credits for the queue. Possible rule for the software calculations of the HiCredit parameter:
HiCredit = 0x80000000 + MAX_TPKT_SIZE * Percentage Link Utilization * 0x7736

Non-negative credits - Credits are equal or greater than 0x80000000.

7.5.2.7.1 Credit Base Shaping and Qbv

The following rules describe the credit accumulation when Qbv is enabled as well:

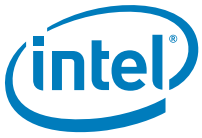
- Aways reduce credits when a packet is transmitted. Either within the Qbv window or outside the Qbv window.
- Always reset positive credits if the transmit queue is empty. Either within the Qbv window or outside the Qbv window.
- Increment negative credits only within the Qbv window.
- Increment positive credits only within the Qbv window and the transmit queue is not empty.
- Freeze the credits outside the Qbv window.

7.5.2.8 Basic Scheduling

Foxville supports the basic scheduling method. In this method, a packet scheduling time is expired if:

SYSTIME **>=** "Scheduling Time", for packet with **inactive** Frst flag in the context descriptor
For packet with **active** Frst flag the above condition should apply and the packet must be the first one in the scheduling window (see Qbv section for description of scheduling window)

The "Scheduling Time" is described in Section 7.2.2.2.3 as: LaunchTime + BaseT + StQT - ~~Global_~~GTxOffset.
It is software responsibility to set the BaseT and StQT registers upfront before initiating the transmit packet with its LaunchTime in the transmit descriptor.
The scheduling time defines a unique time that is expires only one time. The BaseT register is 64 bit as the size of the SYSTIM register while the StQT and the LaunchTime are 30 bit offset relative to the BaseT. Note that this behavior is NOT as its predecessor, on which the scheduling time defines only the sub second units which can be expired multiple times after any whole number of seconds.



If there is a need for repetitive scheduling ~~(~~of whole seconds (or any other repetitive rate), it is required to use the Qbv logic as described in Section 7.5.2.9.

7.5.2.8.1 Basic Scheduling - Parameters

QBV Time Devision registers:

BaseT: The "base time" for the transmit scheduling.

QbvCycleT and QbvCycleT_S: Should be set to zero.

Global Parameters in the **TQAVCTRL** register:

TRANSMIT_MODE: This flag selects between legacy arbitration or TSN mode. The Transmit mode must be set to TSN for any scheduling method.

Sch_Timer_Sel: This field selects which of the 4 x 1588 timers is used to schedule the traffic.

PREEMPT_ENA: Enabling preemption is orthogonal to the scheduling method. Preemption can be enabled only if it is supported by the network and the link partner.

Queue Context Parameters.

TXQCTL.Queue_Mode: Each queue is set to one of 2 modes of operation: with or without Launch time

TXQCTL.Strict_Cycle and TXQCTL.Strict_End: Should be cleared.

StQT[n]: Relative scheduling offset to BaseT.

EndQT[n]: Should be cleared.

7.5.2.9 Time Aware Shaper - 802.1Qbv

Foxville supports the Time Aware Shaper scheme (802.1Qbv). The time is divided to QbvCycleT periods and the beginning of the cycle is indicated by the BaseT register. Each transmit queue has a permitted transmission window within the cycle defined by StQT[n] and EndQT[n] as illustrated in the Figure 7-19 below.

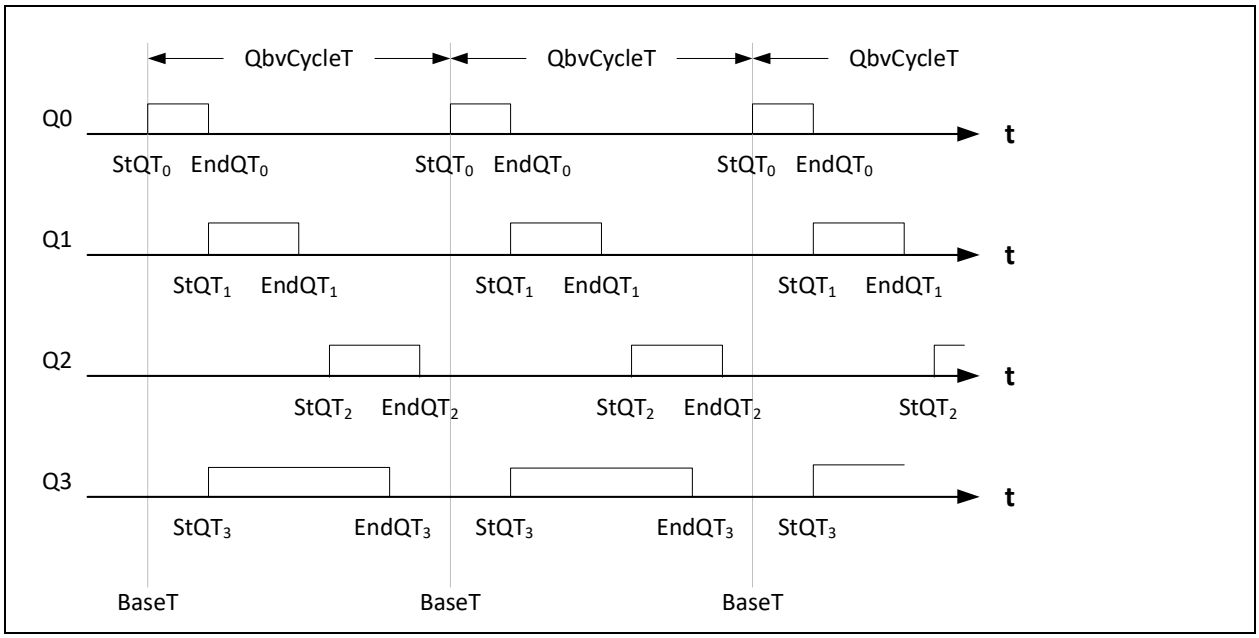


Figure 7-19. 802.1Qbv Transmission Windows

7.5.2.9.1 Time Aware Shaper - Parameters

QBV Time Devision registers:

BaseT: The "base time" for the transmit scheduling.

QbvCycleT:

— **Qbv_Cycle_Time and QbvCycleT_S**: Defines the Qbv cycle time in nsec units. During run time, the QbvCycleT_S is a shadow image of the Qbv_Cycle_Time.

Global Parameters in the **TQAVCTRL** register:

TRANSMIT_MODE: This flag selects between legacy arbitration or TSN mode. The Transmit mode must be set to TSN to enable the Qbv functionality.

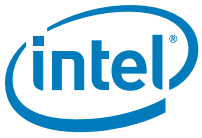
Sch_Timer_Sel: This field selects which of the 4 x 1588 timers is used to schedule the traffic.

PREEMPT_ENA: Enabling preemption is orthogonal to Qbv mode of operation. Preemption can be enabled only if it is supported by the network and the link partner.

Queue Context Parameters in the TXQCTL registers Associating the queues to the QBV parameters.

Queue_Mode: Each queue is set to one of 2 modes of operation: with or without Launch time

Strict_Cycle: When this flag is set, packets from the queue can be transmitted only if they are expected to be completed before the end of the Qbv cycle. This flag is recommended to be set when scheduling scheme is Qbv. It can be cleared if Qbv mode is used only as a mechanism to auto increment the BaseT register each whole second.



Strict_End: When this flag is set, packets from the queue can be transmitted only if they are expected to be completed before the window of the queue is ended.

StQT[n]: The Start Time registers (one per each queue, n=0...3). These registers define the starting of the transmission window of the queue relative to BaseT.

EndQT[n]: The End Time registers (one per each queue, n=0...3). These registers define the completion time of the transmission window of the queue relative to BaseT.

7.5.2.9.2 Time Aware Shaper for whole Packets Transmission - Flow

Qbv scheduling is enabled by setting the Qbv_Cycle_Time and QbvCycleT_S to non-zero value. The transmit scheduling from all queues are relative to the BaseT register. Initialization of the BaseT for the very first Qbv cycle is described in Section 7.5.2.9.3.2. Then the BaseT is updated by QbvCycleT on the same clock that the scheduling timer is expected to cross or being equal to BaseT + QbvCycleT.

Packet transmission is enabled if all the conditions below are met:

1. The whole packet is fetched to the transmit packet buffer.
2. If credit based shaping is assigned to the queue, then transmission is enabled only if the credit is non-negative.
3. Transmission can be initiated only within the Qbv window (the StQT is expired and the EndQT is not yet expired).
4. On top of it, If the Strict_End flag in the TXQCTL register is set as well, transmission is enabled only if the whole packet is expected to be completed before the end of the transmission window.
5. On top of it, transmission is enabled only if it is expected also that the whole packet is ended before the completion of the Qbv Cycle (BaseT + QbvCycleT).

Note: The above rules 5 and 6 do not take into account possible distortions of the completion time due to timer adjustment. Strict maintaining of rule 5 is subjected to the use case. Strict maintaining of rule 6 is a must have for proper device functionality. It is software responsibility keeping large enough gap between the EndQT of all active queues and the end of the Qbv cycle. This gap should be larger than the largest time adjustment in a single cycle.

7.5.2.9.3 Time Aware Shaper for Preempted Fragments - Flow

Non-Express traffic can be preempted in the middle by Express packets. Before preemption happened, the transmission of the Non-Express packet could start only if all conditions described in Section 7.5.2.9.2 were met. Once the packet is preempted it would resume transmission (at the end of the Express packet) regardless of any Qbv windowing and Qav credits boundaries. It is implemented this way in order to avoid possible blocking of other Non-Express queues (until the transmission of this preempted packet is completed).

7.5.2.9.3.1 Packet Data Fetch Time

Packet data fetch is scheduled for TSN queue if enabled by the Data_Fetch_TIM flag in the TXQCTL[n] registers. In this case, the data is fetched not sooner than the transmission time minus Fetch_Tim_Delta parameter in the TXQCTL[n] registers. Calculating the fetch time in Qbv mode might be tricky as it might fall in the current Qbv cycle or the previous cycle. The fetch time is given by the following equation:

BaseT + StQT + LaunchTime - Fetch_Tim_Delta, if StQT + LaunchTime >= Fetch_Tim_Delta
Fetch Time =
BaseT + QbvCycleT + StQT + LaunchTime - Fetch_Tim_Delta, if StQT + LaunchTime < Fetch_Tim_Delta

7.5.2.9.3.2 Initialization the Qbv Base and Cycle Time Registers

Software can initialize the BaseT register in one of two options: Explicit or Autonomous setting.

Explicit Setting option:

1. Software: Set the Shadow_QbvCycle in the QbvCycleT_S register to the desired cycle time.
2. Software: Enable Qbv by setting the Qbv_Cycle_Time to the desired cycle time in the QbvCycleT register.
3. Software: Read the SYSTIM registers (used for the scheduling) and set the BaseT to a specific time in the future that scheduling should start. Software writes first the high BaseT high register and then the BaseT low register. The Copy_H flag and the Auto_Set flag in the BaseT low register should be cleared.
4. Device response: After setting the BaseT low register the device starts updating the BaseT by the Qbv_Cycle_Time whenever the SYSTIM crosses the current BaseT + Qbv_Cycle_Time.

Autonomous Setting option:

This option is useful when the Qbv Cycles are aligned to whole seconds. in this case Foxville can auto sets the BaseT register to the closest time to the current time. It saves the need from the software reading the current time registers in order to set the BaseT registers. The steps below described the software hardware flow for the autonomous setting option:

1. Software: Set the Shadow_QbvCycle in the QbvCycleT_S register as listed below in Table 7-63.
2. Software: Enable Qbv by setting the Qbv_Cycle_Time to the desired cycle time in the QbvCycleT register.
3. Software: Set the BaseT low register to 0xC0000000 (clearing the BaseT low value and setting the Copy_H and the Auto_Set flags).
4. Device response: Auto-correct the Shadow_QbvCycle to the value of the Qbv_Cycle_Time and set the BaseT registers to the "closest" time to SYSTIM as follow:
   Case A = Plus1 flag in the QbvCycleT_S register is cleared to '0': BaseT gets the time value on which the current Qbv cycle is still in progress. Meaning, BaseT = SYSTIM high + init BaseT low in step 3 + 'N' * QbvCycleT, while 'N' is the maximum index that results with BaseT <= SYSTIM.
   Case B = Plus1 flag in the QbvCycleT_S register is set to '1': BaseT gets the time value on which the next Qbv cycle should start. Meaning, BaseT = the above value plus one QbvCycleT.
   This process takes around 100nsec in the 2.5G product and 120nsec in the 1G product.
   For Qbv functionality the Plus1 option should be used. For simplified scheduling on which the Qbv logic is used just to auto-increment the BaseT register either Plus1 or no Plus1 options can be used.
5. Device response: From this point, time base scheduling is enabled and the device updates the BaseT by the Qbv_Cycle_Time whenever the SYSTIM is about to cross the current BaseT plus Qbv_Cycle_Time (a clock time before it should happen).
6. Software can program the transmit queues and bump their tails following step 3 without waiting for a completion of step 4. It is possible since it is guaranteed that by the time the transmit queues will require the BaseT register it will be already valid.

Table 7-63. Permitted Qbv Cycle Times and its Matched Shadow_QbvCycle setting

Qbv Cycle	matched Shadow_QbvCycle	Qbv Cycle	matched Shadow_QbvCycle	Qbv Cycle	matched Shadow_QbvCycle
31.25us	512ms	1ms	512ms	62.5ms	500ms
62.5us	512ms	2ms	512ms	125ms	500ms
125us	512ms	4ms	512ms	250ms	500ms
250us	512ms	8ms	512ms	500ms	500ms
500us	512ms	31.25ms	500ms	1s	1s

7.5.2.9.3.3 The First Packet on Each Qbv Cycle

Transmit queues that should obey the Qbv scheduling might have packets that belong to multiple Qbv cycles. Software should indicate to the hardware the packets that should be the first ones in each Qbv cycle. It is done by setting the Frst flag in the transmit context descriptor.

Handling packets with **inactive frst flag**: Transmit scheduling is enabled when the time is larger than the scheduling time.

Handling packets with **active Frst flag**: Is the same as above plus the packet must be the first whole packet in the scheduling window. Note that if a previous packet starts before the beginning of this Qbv window and it is completed in this current Qbv window, the packet with an active Frst flag can still start in this window. It is the same rule for possible packet that starts in a previous Qbv window and some of its fragments continue in this Qbv window.

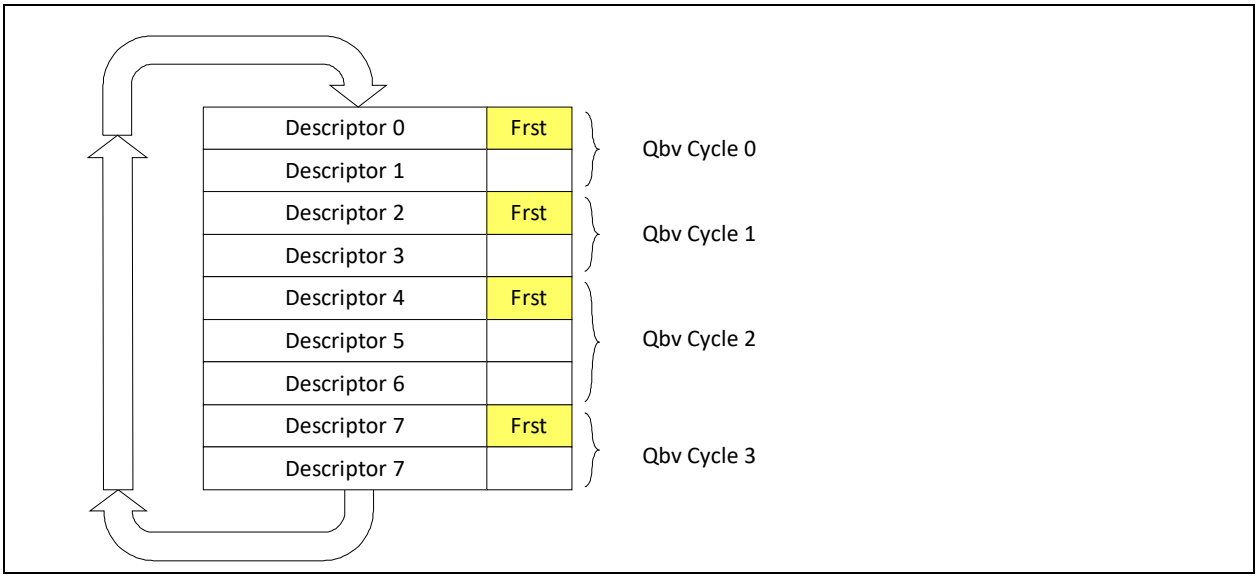


Figure 7-20. Indicating the First Packet on Each Qbv Cycle

7.5.3 Preemption and Interspersing Express Traffic (802.3br)

Please refer to Section 7.5.3.1 for preemption / interspersing (802.1Qbr) functionality.

7.5.3.1 Preemption - 802.3br

Foxville supports an option for Interspersing Express packets in the middle of non-Express packets. This option optimizes link utilization while keeping accurate transmission time of the Express traffic. Foxville categorizes the traffic type to be Express or non-Express according to the transmit queues settings. Each queue is set by the TXQCTL.Preemptable flag to one of the following two options: Preemptive (eTXQ) for EXPRESS queue or Preemptable (pTXQ) for non-EXPRESS queue. Setting a queue as eTXQ enables its packets to have preemption rights on packets originated from queues set as pTXQ. Packets from the pTXQ's might be preempted in the middle if a packet from an eTXQ should be transmitted. In order to enable the preemption scheme, the eTXQ's must be set with higher priority level than pTXQ's by the TxQ_Priority flag in the TxARB register. This chapter and its sub-sections describe the transmit flow intercepting EXPRESS traffic in the middle of non-EXPRESS traffic. Breaking the non-EXPRESS packets into multiple fragments, as well as the receive flow gathering together the multiple fragments back to whole packets.

7.5.3.1.1 mPacket Format

Interspersing Express Traffic introduces a new term called "mPacket" that stands for "Merge Packet". The mPacket differs from the legacy 802.3 packet format by a modified header and trailer as shown in Figure 7-21 Table 7-64 and Table 7-65. mPackets can be enabled only if both link partners support it.

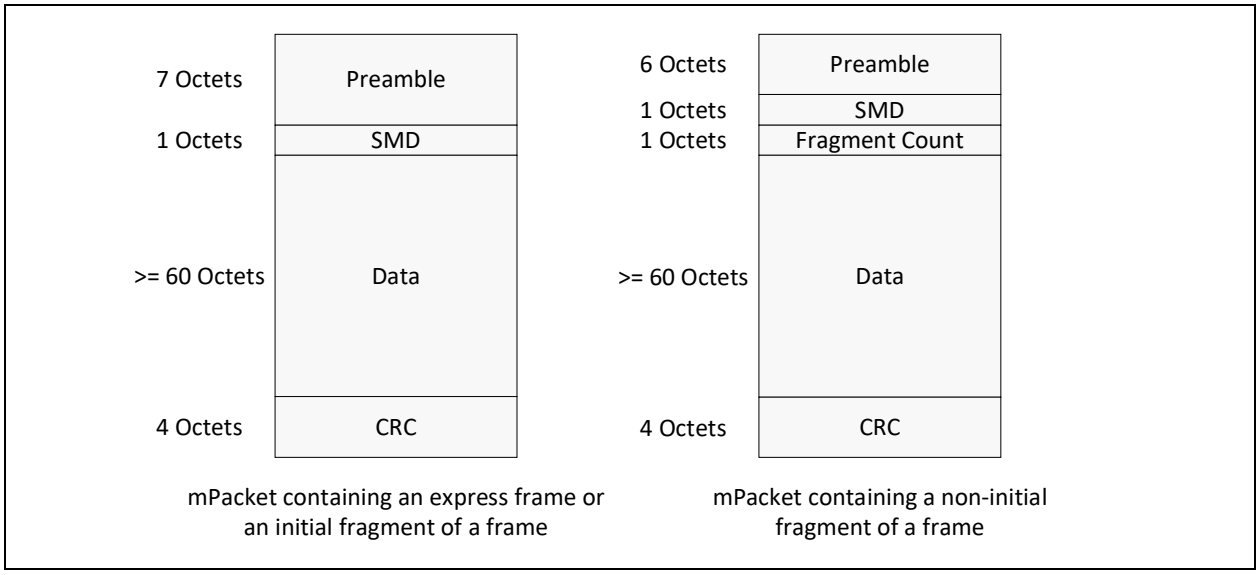
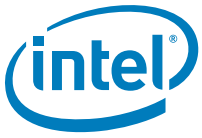


Figure 7-21. mPacket Format

Preamble: The octets of the preamble equals to 0x55. It is 7 octets for an initial fragment or whole packet and 6 octets for non-initial fragments.

SMD - Start mPacket Delimiter (shown in Table 7-64**)**: The value of the SMD indicates whether the mPacket contains an express frame, the initial fragment of a preemptable frame, or any of non-initial fragments of a preemptable frame.



Fragment Count: The fragment count is a modulo-4 counter that is auto-incremented on each fragment of the preemptable frame. The fragment count is set to zero at the initial non-first fragment of the mPackets and it is incremented for each fragment. Encoding of the fragment count is shown in Table 7-65.

Frame Count: The frame count is embedded in the SMD. it is a modulo-4 counter that is auto-incremented on each preemptable frame.

CRC / mCRC: The CRC field contains a cyclic redundancy check (CRC) for the mPacket data and an indication if this is the last fragment of the mPacket. It equals to the standard CRC in the last fragment of the mPacket or whole packet. In any other non-last fragment, it equals to the standard FCS of all the mPacket's bytes including this fragment XORed with 0x0000FFFF instead of XOR with 0xFFFFFFFF. Verify and Response packets also carry the mCRC fiels used to verify that the channel supports it.

Table 7-64. SMD Encoding

mFrame Type	Notation	Frame Count	SMD Encoding
Verify Frame	SMD-V	-	0x07
Response Frame	SMD-R	-	0x19
Express Frame	SMD-E (**SFD**)	-	**0xD5**
Preemptable Frame Start	SMD-S0	0	0xE6
	SMD-S1	1	0x4C
	SMD-S2	2	0x7F
	SMD-S3	3	0xB3
non-Initial Fragment	SMD-C0	0	0x61
	SMD-C1	1	0x52
	SMD-C2	2	0x9E
	SMD-C3	3	0x2D

Table 7-65. Fragment Count Encoding

Fragment Count value	Fragment Count Encoding
0	0xE6
1	0x4C
2	0x7F
3	0xB3

7.5.3.1.2 Transmit Operation When Preemption is Enabled

Transmit processing unit accepts packets from eTXQ's and pTXQ's. The transmit unit preempts a preemptable frame when an eTXQ has a frame to transmit and all the following conditions are met (the preemption transmit unit state machine is illustrated in Figure 7-22):

- The resulted fragment of the preemptable mPacket is at least the minimum fragment size as dictated by the TQAVCTRL.MIN_FRAG parameter.
- Preemption only at boundaries of whole number of 4 octets
- There are at least 64 octets remaining to be transmitted

Preemption nesting is forbidden: It means that if a packet from any pTXQ is preempted, packets from other pTXQ's cannot start as long as the preempted packet is completed.

MIN_FRAG: Foxville requires that the transmitted fragments of the mPacket is at least MIN_FRAG octets.

Hold Request: The hold request is set when one of the eTXQ's is associated to a QBV timer on which the QBV time zone starts.

Active Queue: Means that a packet in the queue is ready to be transmitted in the Tx packet buffer.

NO_QAV_DELAY setting option: When the NO_QAV_DELAY is active the impact of the Hold Request or Active eTXQ start earlier than the its actual time so the transmission from the pTXQ's will not invade to the eTXQ's time zone.

TXQCTL.Strict_End setting option: When the Strict_End is active then:

- Transmission from eTXQ is enabled only if the whole packet fits within the Qbv window (if enabled).
- Transmission from pTXQ is enabled only if the whole packet or fragment fits within the Qbv window (if enabled) or only if the whole packet or fragment can be completed before a packet from other eTXQ's is scheduled.

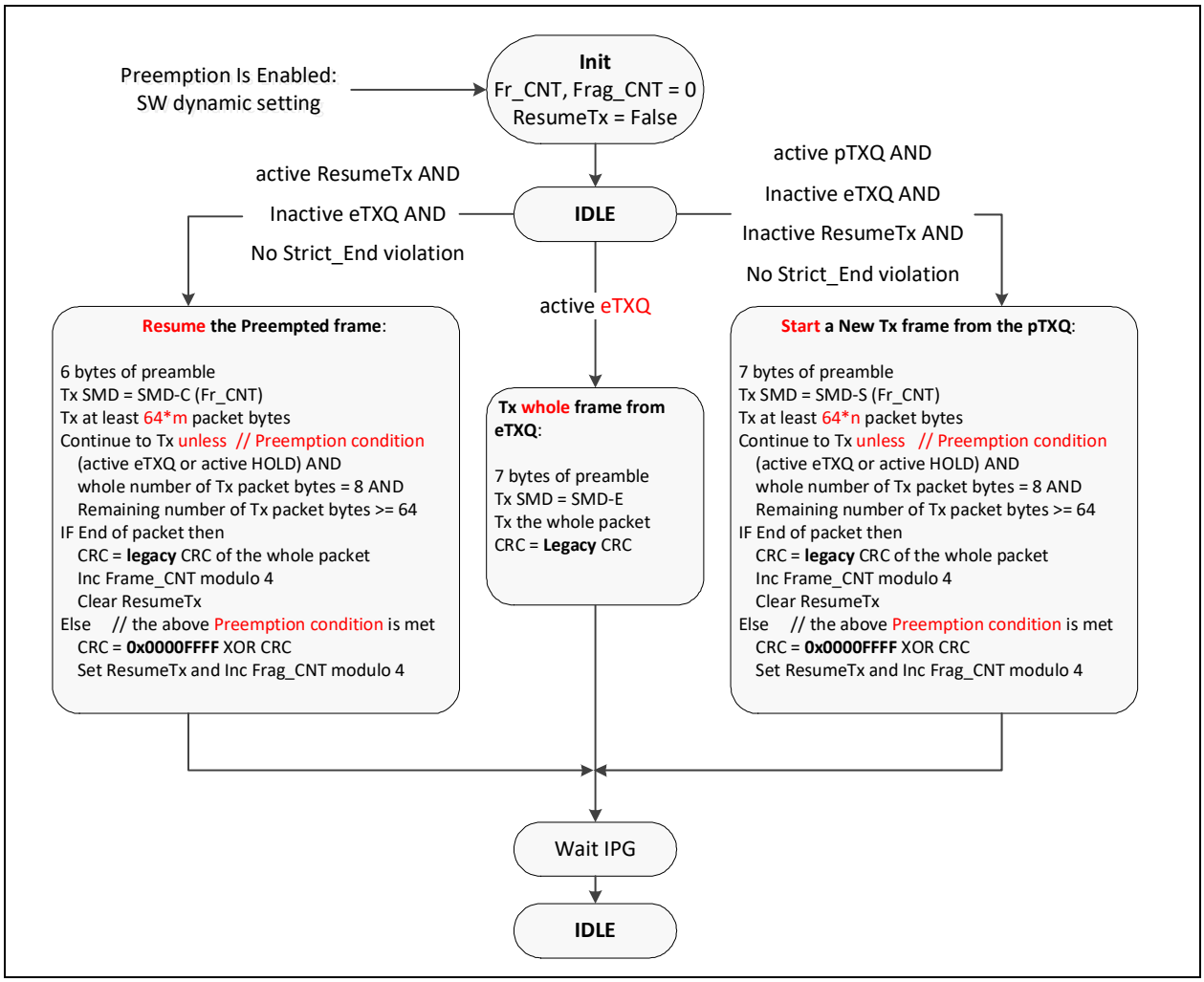
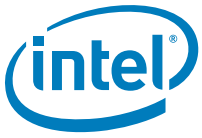


Figure 7-22. Transmit Unit - Preemption State Machine



7.5.3.1.3 Receive Operation When Preemption is Enabled

The receive unit re-assemble the received fragments back to whole packets. It classifies the incoming mPackets as whole packets or packets fragments that should be re-assembled. Classification is done based on the SMD, Fragment Count and the CRC. Foxville categorizes received packets by SMD only when preemption is enabled by the TQAVCTRL.PREEMPT_ENA. Regardless, Verify packets and Response packets are always processed correctly. The receive state machine is illustrated in Figure 7-23 and the receive exceptions are listed in the Table 7-66:

SMD indication: An SMD containing either SMD-V or SMD-R or SMD-E (SFD) indicates a reception of whole packets. An SMD equals to SMD-S indicates the beginning of mPacket that might need to be re-assembled. An SMD equals to SMD-C indicates a non-first fragment that should be combined with previous fragment(s). The SMD encodes also the **Frame Count** field described below.

CRC indication: The CRC of a whole packets as well as the last fragment of a fragmented packet equals the standard FSC. The CRC of non-last fragment equals to the standard FSC of all received fragments XORed with 0x0000FFFF.

Fragment Count: The mPacket header contains also the Fragment Count parameter. Foxville re-assembles fragmented mPackets only when these fragments are received in order. If the fragments are received out of order the mPacket is discarded and it is counted by the PREEMT_OOOP_CNT.

Packet integrity check: Note that the device accumulates the L2 CRC and L4 integrity residuals between consecutive fragments for the sake integrity check.

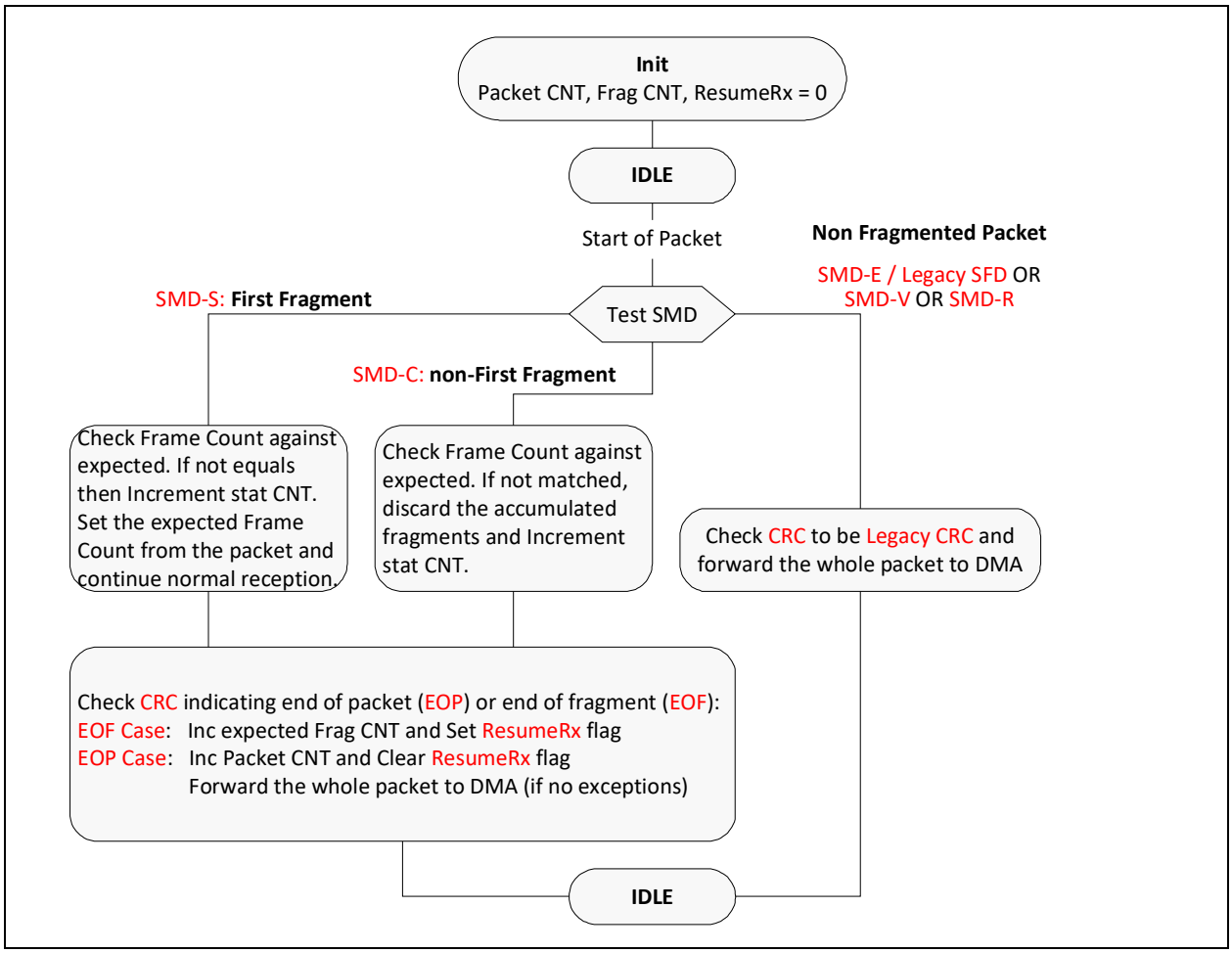
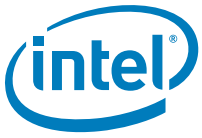


Figure 7-23. Preemption Receive Unit - Packet Re-assembly State Machine

Table 7-66. Preemption Receive Exceptions

Exception	Action	Statistic Counter
Rx SMD-C AND NOT ReumeRx	Discard Packet and GoTo IDLE state	PRMEXCPRCNT.OOO_SMDC
Rx SMD-C AND wrong Frame CNT	Discard Packet and GoTo IDLE state (1)	PRMEXCPRCNT.OOO_FRAME_CNT
Rx SMD-C AND wrong Frag CNT	Discard Packet and GoTo IDLE state (1)	PRMEXCPRCNT.OOO_FRAG_CNT
Rx SMD-S AND ReumeRx	Discard the packet and GoTo IDLE state (1)	PRMEXCPRCNT.MISS_FRAME_FRAG_CNT
Rx SMD-S AND wrong Frame CNT	Process the packet as a good packet	
Comment 1: Discard also any accumulated received fragments		

7.5.3.1.4 Receive Packet Buffers When Preemption is Enabled



When preemption is enabled high priority traffic is assigned a classification of express traffic while the rest is assigned as preemptive traffic. Packets from the express traffic can preempt in the middle packets belonging to the non-express traffic. Because of its high priority, it is desired that packets belonging to the express traffic are posted to the host sooner than the non-express packets and at higher priority. Furthermore, express packet that preempt non-express packet in the middle, should be posted to the host as soon as it is completed without waiting for the preempted packet to be completed.

Addressing the above requirement, the receive packet buffers in Foxville can be divided to 2 x buffers (by the RXPBSIZE_EXP and RXPBSIZE_EXP fields in the RXPBSIZE register). One buffer is allocated to the express traffic and the other one is allocated to the non-express traffic. Completed packets in the "express buffer" take always precedence (strict priority) over packet in the other buffer.

Assigning a unique receive queue to the express traffic will guarantee higher priority inside Foxville.

If a unique queue is not assigned to the express traffic it is not always guaranteed that these packets gets higher priority. If completed packets of the shared queue between express and non-express traffic are accumulated in Foxville, these packets are posted to the host on the order they were completed regardless of its traffic type. There are two possible cases that completed packets are accumulated:

1. The host is slower than the network
2. The queue is programmed with larger write threshold than 0

7.5.3.1.5 Receive Packet Classification when Preemption is Enabled

When preemption is enabled packet classification is based ONLY on the first fragment of the packets. In some applications the "interesting" content of the packet for classification is contained in the first 64 bytes. These applications may care only on L2 packets or IPv4 packets with no IP options and no L4 layer options. Some applications may care about a longer portion of the packets. It may include IPv6 packets. Some applications may care about even longer headers than 128 bytes. No matter the "interesting" header size, it must be contained in the first fragment of the packets if it is needed to be part of the classification process. Therefore, the software should negotiate a minimum preemption fragment size that is large enough to contain the "interesting" header of the packets. The minimum fragment size that the device uses during transmission is configured by the TQAVCTRL.MIN_FRAG parameter.

7.5.3.1.6 Preemption and Wake Up functionality

When preemption is not enabled, all packets start with the legacy SFD. When preemption is enabled, express packets start with the legacy SFD and non-express packets start with SMD-S or SMD-C. When preemption is enabled it is expected that wake up packets are received only by non-express packets identified by SMD-S or SMD-C.

7.5.3.1.7 Preemption Init Flow

All nodes in the network start their operation with no preemption option. Stations negotiate the preemption capability, including the minimum supported fragment size and only then enables the preemption scheme. Enabling preemption, software should first set the TXQCTL.Preemptable to eTXQ in those queues that should carry express traffic (it can be done at queue initialization or during run time). Then software should enable the transmit preemption scheme by setting the global PREEMPT_ENA flag. Setting will take affect at whole packet boundaries.

7.5.3.1.8 Debug Registers and Statistic Counters

QBV_HEADER: provides an indication of the last transmitted header of the last transmitted preemptable packet. It includes the following fields: SMD index (3 bits) and SMD encoding (8 bits); Fragment count index (2 bits) and encoding (8 bits). The number of fragments in the last fragmented packet (8 bit field).

QBV_ppPacket_TCNT: counts the total number of good transmitted preempted packets.

QBV_ppPacket_RCNT: counts the total number of good received preempted packets.

QBV_pPacket_TCNT: counts the total number of good transmitted non-preempted preemptable packets.

QBV_pPacket_RCNT: counts the total number of good received non-preempted preemptable packets.

QBV_ePacket_TCNT: counts the total number of good transmitted non-preemptable packets.

QBV_ePacket_RCNT: counts the total number of good received non-preemptable packets.

QBV_EXCEP.OOO_SMDC: See description in Table 7-66.

QBV_EXCEP.OOO_FRAME_CNT: See description in Table 7-66.

QBV_EXCEP.OOO_FRAG_CNT: See description in Table 7-66.

QBV_EXCEP.MISS_FRAME_FRAG_CNT: See description in Table 7-66.

7.5.4 TSN Configuration

Context Descriptor Configuration

LaunchTime (30 bits). A relative offset of the packet transmission time. See Section 7.2.2.2.3 for details.

Global TSN configuration:

TXPBSIZE.Tx0pbsize/Tx1pbsize/Tx2pbsize/Tx3pbsize (Tx packet buffer size assignment): Recommended configuration when using 4 queues would be 5KB per each of the 4 packet buffers.

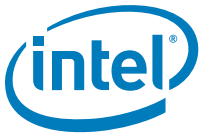
RXPBSIZE.Rxpbsize (Rx packet buffer size assignment): Refer to the setting rule defined in Section 4.7.9, taking into account 24 KB for the transmit packet buffer.

TxARB Register: Setting the transmission priority of the host and MNG queues. Express queues must be set with higher priority than Preemptable queues. If the time window of TSN queues and BE queues overlap, it make sense to set the TSN queues with higher priority than the BE queues.

TQAVCTRL.TRANSMIT_MODE (Transmit mode configuration): The transmit mode should be set to "TSN" mode on which there is a dedicated transmit packet buffer per each of the four queues.

TQAVCTRL.DATA_FETCH_ARB (Data fetch arbitration configuration): Can be set to Round Robin or Most empty.

TQAVCTRL.MIN_FRAG (Minimum length of the transmitted fragments of a preemptable mPacket): It is measured in 64 bytes units.



TQAVCTRL.PREEMPT_ENA (Enables the transmit preemption state machine): This is a dynamic parameter that can be set while the transmit unit is active. Setting takes affect at whole packet boundaries.

DTXMXPKTSZ.MAX_TPKT_SIZE: When 4 x transmit packet buffers are enabled (TRANSMIT_MODE = TSN) the maximum transmit packet size should be decreased to 0x19 (1600 bytes).

Per Queue TSN configuration in the TXQCTL[n] registers:

QUEUE_MODE (Launch time enable option)

Strict_Cycle and **Strict_End**: These parameters enables transmission according to the expected duration of the packets relative the the Qbv cycle.

Report_TIMER_SEL: Each transmit queue is associated to one of the 1588 timers by the Report_TIMER_SEL flag. This association impact the reported DMA completion time. Transmit packet sampling is not affected by this flag but rather by explicit setting of the 1588_Sel flag in the transmit descriptor.

Preemptable: Each transmit queue can be set as Preemptive - eTXQ for express traffic or Preemptable - pTXQ for non-Express traffic. The pTXQs can be preempted by the eTXQs when preemption is enabled by the TQAVCTRL.PREEMPT_ENA.

DATA_FETCH_TIM (Rx Data fetch Time Valid configuration): The DATA_FETCH_TIM flag enables the data fetch time relative to the transmission time.

FETCH_TIM_DELTA (Fetch Time Delta configuration): The time to reduce from Launch time to make a TSN queue packet eligible for fetch. This parameter is meaningful only if the DATA_FETCH_TIM flag is set.

Additional Per Queue TSN related configuration:

TXDCTL.Priority: It can be used to prioritize TSN queues over BE queues or prioritize Express queues over Preemptable queues.

Time Aware Shaper - Parameters: See Section 7.5.2.9.1.

7.5.5 PTP Packet Structure - Background

The time sync implementation supports both the 1588 V1 and V2 PTP frame formats. The V1 structure can come only as UDP payload over IPv4 while the V2 can come over L2 with its Ethertype or as a UDP payload over IPv4 or IPv6. The 802.1AS uses only the layer 2 V2 format. The PTP frame formats over L2 and over UDP are listed in the Table 7-67 and Table 7-68. The PTP V1 and V2 message formats are listed in the Table 7-69 followed by SYNC packet format in Table 7-70. Table 7-71 and Table 7-72 list the relevant fields that identify the PTP message that are the *Control* field for V1 message and the MessageType field for V2 message. Then, Table 7-73 lists the device settings required to identify the PTP packets.

Table 7-67. PTP Message Over Layer 2

Ethernet (L2)	VLAN (Optional)	PTP Ethertype	PTP message

Table 7-68. PTP Message Over Layer 4

Ethernet (L2)	IP (L3)	UDP	PTP message

Table 7-69. V1 and V2 PTP Message Header

Offset in Bytes	V1 Fields	V2 Fields	
Bits	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	
0	***versionPTP***	transport Specific[1]	***messageType***
1		Reserved	***versionPTP***
2	version Network	message Length	
3			
4	Subdomain	domain Number	
5		Reserved	
6		flags	
7			
8		correctionField	
9			
10			
11			
12			
13			
14			
15			
16		reserved	
17			
18			
19			
20	message Type	Source Port Identity	
21	Source communication technology		
22	Sourceuuid		
23			
24			
25			
26			
27			
28	source port id		
29			
30	***sequenceId***	***sequenceId***	
31			
32	***control***	control	
33	reserved	Log Message Interval	
34	flags	**PTP message body. The PTP header plus its message body must be al least 36 bytes long to be recognized as a PTP message.**	
35			

1. Should be all zero.

Note: Only the fields with the bold italic format colored red are of interest to the hardware.

Table 7-70. SYNC Message Structure

Offset in Bytes	Length in Bytes	Fields
0	34	Message Header (as listed in Table 7-69).
34	10	SYNC Timestamp. On 1-step transmission the timestamp is inserted by the hardware: The first 2 bytes equals to *SYSTIMTM.STM* while its MS byte is first and its LS byte is last The next 4 bytes equals to *SYSTIMH* while its MS byte is first and its LS byte is last The next 4 bytes equals to *SYSTIML* while its MS byte is first and its LS byte is last

Table 7-71. Message Decoding for V1 (Control Field at Offset 32)

Enumeration	Value
PTP_SYNC_MESSAGE	0
PTP_DELAY_REQ_MESSAGE	1
PTP_FOLLOWUP_MESSAGE	2
PTP_DELAY_RESP_MESSAGE	3
PTP_MANAGEMENT_MESSAGE	4
reserved	5–255

Table 7-72. Message Decoding for V2 (MessageType Field at Offset 0)

MessageType	Message Type	Value (hex)
PTP_SYNC_MESSAGE	Event	0
PTP_DELAY_REQ_MESSAGE	Event	1
PTP_PATH_DELAY_REQ_MESSAGE	Event	2
PTP_PATH_DELAY_RESP_MESSAGE	Event	3
Unused	Event	4-7
PTP_FOLLOWUP_MESSAGE	General	8
PTP_DELAY_RESP_MESSAGE	General	9
PTP_PATH_DELAY_FOLLOWUP_MESSAGE	General	A
PTP_ANNOUNCE_MESSAGE	General	B
PTP_SIGNALLING_MESSAGE	General	C
PTP_MANAGEMENT_MESSAGE	General	D
Unused	General	E-F

Foxville identifies both L2 and L4 PTP packets for timestamp sampling and defining a specific receive queue as listed in the Table 7-73.

Table 7-73. Enabling Receive Timestamp

Functionality	Register	Field	Setting Options
Enable receive timestamp	TSYNCRXCTL	En	En = 1b (must be set in all the following options).
Sampled V1 Control value	TSYNCRXCFG	CTRLT	The CTRLT defined the recognized V1 Control field. This field must be defined if V1 packets recognition is required.
Sampled V2 MessageType value	TSYNCRXCFG	MSGT	The MSGT defined the recognized V2 MessageType field. This field must be defined if V2 packets recognition is required.

Table 7-73. Enabling Receive Timestamp

Functionality	Register	Field	Setting Options
Enable all packets for timestamp	TSYNCRXCTL	Type	Type equals to 100b enables sampling all packets. Useful only when posting the timestamp to the packet buffer in host memory, enabled per queue by the SRRCTL[n].Timestamp.
Enable L2 1588 packets for timestamp sampling	TSYNCRXCTL	Type	Type equals to 000b or 010b enable V2 packets with MessageType equals to MSGT as well as DELAY_REQ and DELAY_RESP packets. Type equals to 101b enable all V2 packets with Message Type bit 3 zero (means any event packets)
	ETQF[n]	EType Filter enable	The EType on one of the enabled ETQF registers (Filter enable is '1') should be set to the 1588 EtherType (equals to 0x88F7)
Enable 1588 packets over UDP for timestamp sampling	TSYNCRXCTL	Type	Type equals to 001b enables V1 packets with Control field equals to CTRLT parameter Type equals to 010b enables V2 packets with MessageType fields equals to MSGT parameter as well as DELAY_REQ and DELAY_RESP packets. Type equals to 101b enables all V2 packets with Message Type bit 3 zero (which means any event packets)
	TTQF[n]	Protocol 1588 time stamp	Defines a UDP protocol (Protocol field = 17 dec). The "1588 time stamp" flag is active
	IMIR[n]	Destination Port PORT_BP	Define PTP event messages (Destination Port = 319 dec) and the PORT_BP is cleared
Define specific receive queue for the L2 1588 packets	ETQF[n]	Rx Queue Queue Enable	Setting the "Queue Enable" on the same ETQF register as above, the receive queue is defined by the "Rx Queue" field.
Define specific receive queue for 1588 packets over UDP	TTQF[n]	Rx Queue Queue Enable	Setting the "Queue Enable" on the same TTQF register as above, the receive queue is defined by the "Rx Queue" field.

7.6 Statistic Counters

Foxville supports different statistic counters as described in Section 8.18. The statistic counters can be used to create statistic reports as required by different standards. Foxville statistic counters allow support for the following standards:

- IEEE 802.3 clause 30 management – DTE section.
- NDIS 6.0 OID_GEN_STATISTICS.
- RFC 2819 – RMON Ethernet statistics group.
- Linux Kernel (version 2.6) net_device_stats

The following section describes the match between the internal Foxville statistic counters and the counters requested by the different standards.

7.6.1 IEEE 802.3 Clause 30 Management

Foxville supports the Basic and Mandatory Packages defined in clause 30 of the IEEE 802.3 specification. Table 7-74 lists the matching between the internal statistics and the counters requested by these packages.

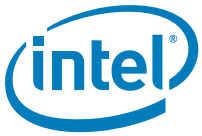


Table 7-74. IEEE 802.3 Mandatory Package Statistics

Mandatory Package Capability	Foxville Counter	Notes and Limitations
FramesTransmittedOK	GPTC	Foxville doesn't include flow control packets.
SingleCollisionFrames	SCC	
MultipleCollisionFrames	MCC	
FramesReceivedOK	GPRC	Foxville doesn't include flow control packets.
FrameCheckSequenceErrors	CRCERRS	
AlignmentErrors	ALGNERRC	

In addition, part of the recommended package is also implemented as listed in Table 7-75.

Table 7-75. IEEE 802.3 Recommended Package Statistics

Recommended package capability	Foxville Counter	Notes and Limitations
OctetsTransmittedOK	GOTCH/GOTCL	Foxville counts also the DA/SA/LT/CRC as part of the octets. Foxville doesn't count Flow control packets.
FramesWithDeferredXmissions	DC	
LateCollisions	LATECOL	
FramesAbortedDueToXSColls	ECOL	
FramesLostDueToIntMACXmitError	HTDMPC	Foxville counts the excessive collisions in this counter, while 802.3 increments no other counters, while this counter is incremented
CarrierSenseErrors	TNCRS	Foxville doesn't count cases of CRS de-assertion in the middle of the packet. However, such cases are not expected when the internal PHY is used. In Foxville this counter is not operational in 100 Mb/s half duplex mode. (HSD 359326)
OctetsReceivedOK	TORL+TORH	Foxville counts also the DA/SA/LT/CRC as part of the octets. Doesn't count Flow control packets.
FramesLostDueToIntMACRcvError	RNBC	
SQETestErrors	N/A	
MACControlFramesTransmitted	N/A	
MACControlFramesReceived	N/A	
UnsupportedOpcodesReceived	FCURC	
PAUSEMACCtrlFramesTransmitted	XONTXC + XOFFTXC	
PAUSEMACCtrlFramesReceived	XONRXC + XOFFRXC	

Part of the optional package is also implemented as described in Table 7-79.

Table 7-76. IEEE 802.3 Optional Package Statistics

Optional package capability	Foxville Counter	Notes
MulticastFramesXmittedOK	MPTC	Foxville doesn't count FC packets
BroadcastFramesXmittedOK	BPTC	
MulticastFramesReceivedOK	MPRC	Foxville doesn't count FC packets
BroadcastFramesReceivedOK	BPRC	

Table 7-76. IEEE 802.3 Optional Package Statistics (Continued)

InRangeLengthErrors	LENERRS	
OutOfRangeLengthField	N/A	Packets parsed as Ethernet II packets
FrameTooLongErrors	ROC + RJC	

7.6.2 OID_GEN_STATISTICS

Foxville supports the part of the OID_GEN_STATISTICS as defined by Microsoft* NDIS 6.0 specification. Table 7-77 lists the matching between the internal statistics and the counters requested by this structure.

Table 7-77. Microsoft* OID_GEN_STATISTICS

OID entry	Foxville Counters	Notes
ifInDiscards;	MPC or MPC + RNBC	It is the MPC if the Drop_En flag is cleared and it is MPC + RNBC if the Drop_En flag is set.
ifInErrors;	RERC	
ifInOctets	GORC / HGORC	The GORC counter report all received octets while the HGORC reports only those octets aimed for the host rather than Host + CSME.
ifHCInOctets;	GORCL/GOTCL	
ifHCInUcastPkts;	GPRC - MPRC - BPRC	
ifHCInMulticastPkts;	MPRC	
ifHCInBroadcastPkts;	BPRC	
ifHCOutOctets;	GOTCL/GOTCH	
ifHCOutUcastPkts;	GPTC - MPTC - BPTC	
ifHCOutMulticastPkts;	MPTC	
ifHCOutBroadcastPkts;	BPTC	
ifOutErrors;	ECOL + LATECOL	
ifOutDiscards;	ECOL	
ifHCInUcastOctets;	N/A	
ifHCInMulticastOctets;	N/A	
ifHCInBroadcastOctets;	N/A	
ifHCOutUcastOctets;	N/A	
ifHCOutMulticastOctets;	N/A	
ifHCOutBroadcastOctets;	N/A	

7.6.3 RMON

Foxville supports the part of the RMON Ethernet statistics group as defined by IETF RFC 2819. Table 7-78 lists the matching between the internal statistics and the counters requested by this group.

Table 7-78. RMON Statistics

RMON statistic	Foxville Counters	Notes
etherStatsDropEvents	*MPC + RNBC*	
etherStatsOctets	*TOTL + TOTH*	
etherStatsPkts	TPR	

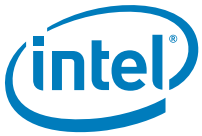


Table 7-78. RMON Statistics (Continued)

RMON statistic	Foxville Counters	Notes
etherStatsBroadcastPkts	BPRC	
etherStatsMulticastPkts	MPRC	Foxville doesn't count FC packets
etherStatsCRCAlignErrors	*CRCERRS + ALGNERRC*	
etherStatsUndersizePkts	RUC	
etherStatsOversizePkts	ROC	
etherStatsFragments	RFC	Should count bad aligned fragments as well
etherStatsJabbers	RJC	Should count bad aligned jabbers as well
etherStatsCollisions	COLC	
etherStatsPkts64Octets	PRC64	RMON counts bad packets as well
etherStatsPkts65to127Octets	PRC127	RMON counts bad packets as well
etherStatsPkts128to255Octets	PRC255	RMON counts bad packets as well
etherStatsPkts256to511Octets	PRC511	RMON counts bad packets as well
etherStatsPkts512to1023Octets	PRC1023	RMON counts bad packets as well
etherStatsPkts1024to1518Octets	PRC1522	RMON counts bad packets as well

7.6.4 Linux net_device_stats

Foxville supports part of the net_device_stats as defined by Linux Kernel version 2.6 (defined in <linux/netdevice.h>). Table 7-79 lists the matching between the internal statistics and the counters requested by this structure.

Table 7-79. Linux net_device_stats

net_device_stats field	Foxville Counters	Notes
rx_packets	GPRC	Foxville doesn't count flow controls - can be accounted for by using the XONRXC and XOFFRXC counters
tx_packets	GPTC	Foxville doesn't count flow controls - can be accounted for by using the XONTXC and XOFFTXC counters
rx_bytes	*GORCL + GORCH*	
tx_bytes	*GOTCL + GOTCH*	
rx_errors	*CRCERRS + RLEC + RXERRC + ALGNERRC*	
tx_errors	*ECOL + LATECOL*	
rx_dropped	N/A	
tx_dropped	N/A	
multicast	MPTC	
collisions	COLC	
rx_length_errors	RLEC	
rx_over_errors	N/A	
rx_crc_errors	CRCERRS	
rx_frame_errors	ALGNERRC	
rx_fifo_errors	*HRMPC* + Sum (RQDPC)	
rx_missed_errors	MPC	
tx_aborted_errors	ECOL	
tx_carrier_errors	N/A	
tx_fifo_errors	N/A	

Table 7-79. Linux net_device_stats (Continued)

net_device_stats field	Foxville Counters	Notes
tx_heartbeat_errors	N/A	
tx_window_errors	LATECOL	
rx_compressed	N/A	
tx_compressed	N/A	

7.6.5 Statistics Hierarchy

The following diagram shows the relations between the packet flow and the different statistic counters.

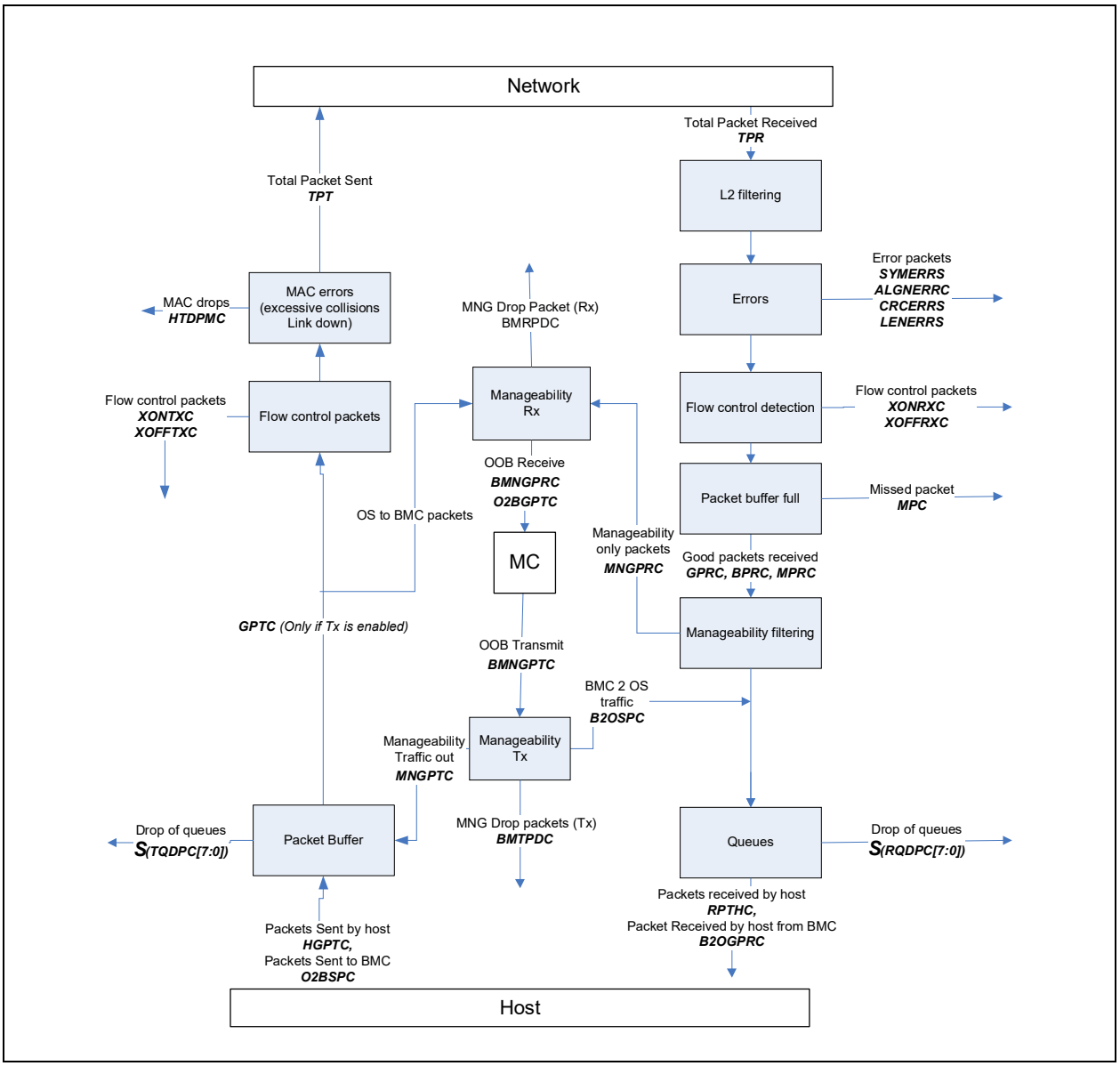


Figure 7-24. Flow Statistics

7.7 Memory Error Correction and Detection

Foxville main internal memories are protected by error correcting code or parity bits. Large memories or critical memories are protected by an error correcting code (ECC). Smaller memories are protected either with an error correcting code (ECC for critical memories) or by parity.

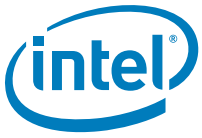
Foxville reports parity errors in the *PEIND* register according to the region in which the parity error occurred (PCIe, DMA, LAN Port or Management). An interrupt is issued via the *ICR.FER* bit on occurrence of a parity error. Parity error interrupt generation per region can be masked via the *PEINDM* register.

Additional per region granularity in parity or ECC enablement, parity error injection and reporting of parity error or ECC parity correction occurrence is supported in the following registers:

1. PCIe region:
   a. The *PCIEERRCTL* and *PCIEECCCTL* registers enable parity checks and ECC parity correction respectively in the various rams in the PCIe region. In addition the registers also enable parity error injection to the PCIe rams for DFT purposes.
   b. The *PCIEERRSTS* and *PCIEECCSTS* registers report parity error and ECC parity correction occurrence in the various rams in the PCIe region. Only parity errors that were not corrected by the ECC circuitry are reported by asserting the *PEIND.pcie_parity_fatal_ind* bit and the *ICR.FER* bit. Parity errors that were corrected by the internal ECC circuit do not generate an interrupt but are logged in the *PCIEECCSTS* register.
2. DMA region:
   a. The *PBECCSTS* register enables ECC parity correction in the various rams in the DMA region. In addition the registers also enable error injection to the DMA rams for DFT purposes.
   b. The *PBECCSTS* register reports occurrence of ECC parity correction events in the various rams in the DMA region. Only parity errors that were not corrected are reported by setting the *PEIND.dma_parity_fatal_ind* bit and the *ICR.FER* bit. Parity errors that were corrected by the internal ECC circuitry don't generate an interrupt but are logged in the *PBECCSTS* register.
3. LAN Port region:
   a. The *LANPERRCTL* register enables parity checks in the various rams in the LAN Port region.
   b. The *LANPERRINJ* register enables error injection to the LAN Port rams for DFT purposes.
4. The *LANPERRSTS* register reports detection of parity errors. The parity errors that were not corrected are reported via the *PEIND.lanport_parity_fatal_ind* bit and the *ICR.FER* bit.Management region:
   a. The *MNGPARCTL* register enables parity checks while the MNGECCCTL register enables ECC parity error correction in the various rams in the Management region.
   b. The *MNGPARINJ* and *MNGECCCTL* registers enable error injection for DFT purposes to the Management rams.
   c. The *MNGPARSTS* register reports detection of parity errors while the *MNGECCSTS* register reports occurrence of ECC parity correction events in the various rams in the Management region. Only parity errors that were not corrected are reported via the *PEIND.mng_parity_fatal_ind* bit and the *ICR.FER* bit. Parity errors that were corrected by the internal ECC circuit do not generate an interrupt but are logged in the *MNGECCSTS* register. In addition to notifying the Host on parity occurrence an interrupt is sent to the processor in the Management block to indicate occurrence of the parity error, so that Firmware recovers from the parity error event.

Notes:

1. An interrupt to the Host is generated on occurrence of a fatal memory error if the appropriate mask bits in the *PEINDM* register are set and the *IMS.FER* Mask bit is set.



2. All Parity error checking can be disabled via the *GPAR_EN* bit in the *Initialization Control Word 1* NVM word (See Section 6.1.1.11) or by clearing the *PCIEERRCTL.GPAR_EN* bit (See Section 8.23.4).
3. If the *PARITY_ERR_RST_EN* bit in the *Common Firmware Parameters 2* NVM word is set, than a Firmware reset is generated by Hardware on occurrence of a parity error.

7.7.1 Software Recovery From Parity Error Event

If a parity error was detected in one of the internal control memories of the DMA, PCIe or LAN port clusters, the consistency of the receive/transmit flow can not be guaranteed any more. In this case the traffic on the PCIe interface is stopped, since this is considered a fatal error.

If a parity error is detected on the Manageability cluster, PCIe traffic is not affected but an internal reset to the Manageability cluster is generated.

Note: An internal Firmware reset is issued, if enabled by the *PARITY_ERR_RST_EN* bit in the *Common Firmware Parameters 2* NVM word, following detection of a parity error in the Management cluster.

To recover from a parity error event software should initiate the following actions depending on the region in which the parity error occurred.

7.7.1.1 Recovery from PCIe Parity Error Event

To recover from a parity error condition in the PCIe region, the software device driver should:

1. Issue a Device Reset by asserting the *CTRL.DEV_RST* bit.
2. wait at least 3 milliseconds after setting *CTRL.DEV_RST* bit before attempting to check if the bit was cleared or before attempting to access any other register.
3. Initiate the master disable algorithm as defined in Section 5.2.3.3.
4. Clear the PCIe parity error status bits that were set in the *PCIEERRSTS* register.
5. Re-initialize the port.

7.7.1.2 Recovery from DMA Parity Error Event

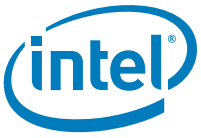
To recover from a parity error condition in the DMA region, the software device driver should issue a software reset by asserting the *CTRL.DEV_RST* bit as specified in Section 4.3.1 and re-initializing the port.

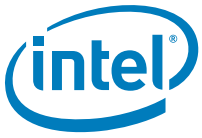
7.7.1.3 Recovery from LAN Port Parity Error Event

To recover from a parity error condition in the LAN port region, the software device driver should take the actions depicted in Section 8.23.12 (*LANPERRSTS* register) according to the ram that failed.

7.7.1.4 Recovery from Management Parity Error Event

To recover from a parity error condition in the Management region if automatic FW reset is not generated (depending on value of *PARITY_ERR_RST_EN* bit in the *Common Firmware Parameters 2* NVM word) Firmware should clear parity error bit in *MNGPARSTS* register after re-initializing u-controller instruction or data RAM if parity error occurred in the NCSI u-controller.





8.0 Programming Interface

8.1 Introduction

This section details the programmer visible state inside Foxville. In some cases, it describes hardware structures invisible to software in order to clarify a concept. Foxville's address space is mapped into four regions with PCI Base Address registers described in Section 9.3.11. These regions are listed in Table 8-1.

Table 8-1. Address Space Regions

Addressable Content	How Mapped	Size of Region
Internal registers, memories and Flash (Memory BAR)	Direct memory-mapped	128 KB + Flash Size [1]
Flash (optional)	Direct memory-mapped	64 KB to 8 MB
Expansion ROM (optional)	Direct memory-mapped	512 KB [2]
Internal registers and memories, Flash (optional)	I/O window mapped	32 bytes[2]
MSI-X (optional)	Direct memory-mapped	16 KB

1. The Flash space in the Memory CSR and Expansion ROM Base Address are mapped to different Flash memory regions. Accesses to the Memory BAR at offset 128 KB are mapped to the Flash device at offset 0x0, while accesses to the Expansion ROM at offset 0x0 are mapped to the fixed Flash region that starts at NVM word address 0x001000. See Section 3.3.3.1. The Expansion ROM region has a fixed provisioned size of 512 KB.
2. The internal registers and memories can be accessed though I/O space indirectly as explained in the sections that follow.

The internal register/memory space is described in the following sections. The PHY registers are accessed through the MDIO interface.

8.1.1 Memory, I/O Address and Configuration Decoding

8.1.1.1 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; refer to Section 9.3.11). Refer to Section 8.1.3 for the appropriate offset for each specific internal register.

8.1.1.2 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; refer to Section 9.3.11). For accesses, the offset from the Memory BAR minus 128 KB corresponds to the physical address within the external Flash device. Memory mapped accesses to the external Flash device are enabled when the value of the *FLBAR_Size* field in the NVM (refer to Section 6.1.1.39) is not 000b.

8.1.1.3 Memory-Mapped Access to MSI-X Tables

The MSI-X tables can be accessed as direct memory-mapped offsets from the base address register (BAR3; refer to Section 9.3.11). Refer to Section 8.1.3 for the appropriate offset for each specific internal MSI-X register.

8.1.1.4 Memory-Mapped Access to Expansion ROM

The Expansion/Option ROM module located in the external Flash (refer to Section 3.3.10.1) can be accessed as a memory-mapped Expansion ROM. Accesses to offsets starting from the Expansion ROM Base address reference the Flash, provided that access is enabled through the *LAN Boot Disable* bit in NVM Initialization Control 3 word, and if the Expansion ROM Base Address register contains a valid (non-zero) base memory address.

8.1.1.5 I/O-Mapped Access to Internal Registers and Memories

To support pre-boot operation (prior to the allocation of physical memory base addresses), all internal registers and memories can be accessed using I/O operations. I/O accesses are supported only if an I/O Base Address is allocated and mapped (BAR2; refer to Section 9.3.11), the BAR contains a valid (non-zero value), and I/O address decoding is enabled in the PCIe configuration.

Note: To enable CSR access via IO address space the *IO_Sup* NVM bit should be set (See Section 6.1.1.25).

When an I/O BAR is enabled by the IO_Sup flag in the NVM, the I/O address range allocated 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 or memory, and then the IODATA register is used as a window to the register or memory address specified by IOADDR:

Table 8-2. IOADDR and IODATA in I/O Address Space

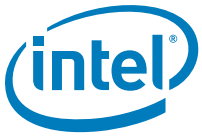
Offset	Abbreviation	Name	RW	Size
0x00	IOADDR	Internal register, internal memory, or Flash location address. 0x00000-0x1FFFF – Internal registers and memories. 0x20000-0xFFFFFFFF – Undefined.	RW	4 bytes
0x04	IODATA	Data field for reads or writes to the internal register, internal memory, or Flash location as identified by the current value in IOADDR. All 32 bits of this register can be read or written to.	RW	4 bytes
0x08 ... 0x1F	Reserved	Reserved.	RO	4 bytes

8.1.1.5.1 IOADDR (I/O Offset 0x00)

The IOADDR register must always be written as a Dword access. Writes that are less than 32 bits are ignored. Reads of any size return a Dword of data; however, the chipset or CPU might only return a subset of that Dword.

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.

Because only a particular range is addressable, the upper bits of this register are hard coded to zero. Bits 31 through 20 cannot be written to and are always read back as 0b.



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.

8.1.1.5.2 IODATA (I/O Offset 0x04)

The IODATA register must always be written as a Dword access when the IOADDR register contains a value for the internal register and memories (for example, 0x00000-0x1FFFC). In this case, writes that are less than 32 bits are ignored.

Reads to IODATA of any size returns a Dword of data. However, the chipset or CPU might only return a subset of that Dword.

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).

Writes and reads to IODATA when the IOADDR register value is in an undefined range (0x20000-0xFFFFFFFC) should not be performed. Results cannot be determined.

Notes: 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, Foxville delays the results through normal bus methods (for example, split transaction or transaction retry).

Because a register/memory 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.

Software should access CSRs via I/O address space or configuration address space but should not use both mechanisms at the same time.

8.1.1.5.3 Undefined I/O Offsets

I/O offsets 0x08 through 0x1F are considered to be reserved offsets with the I/O window. Dword reads from these addresses returns 0xFFFF; writes to these addresses are discarded.

8.1.1.6 Configuration Access to Internal Registers and Memories

To support legacy pre-boot 16-bit operating environments without requiring I/O address space, Foxville enables accessing CSRs via configuration address space by mapping the *IOADDR* and *IODATA* registers into configuration address space. The registers mapping in this case is listed in Table 8-3.

Note: To enable CSR access via configuration address space the *CSR_conf_en* NVM bit should be set (see Section 6.1.1.25).

Table 8-3. IOADDR and IODATA in Configuration Address Space

Configuration Address	Abbreviation	Name	RW	Size
0x98	IOADDR	Internal register or internal memory location address. 0x00000-0x1FFFF– Internal registers and memories. 0x20000-0x7FFFFF – Undefined.	RW	4 bytes
0x9C	IODATA	Data field for reads or writes to the internal register or internal memory location as identified by the current value in IOADDR. All 32 bits of this register can be read or 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 the IOADDR register where:
   a. Bit 31 (*IOADDR.Configuration IO Access Enable*) of the IOADDR register should be set to 1b.
   b. Bits 30:0 of IOADDR should hold the actual address of the internal register or memory being written to.
2. Data to be written is written into the IODATA register.
   — The IODATA register is used as a window to the register or memory address specified by IOADDR register. As a result, the data written to the IODATA register is written into the CSR pointed to by bits 30:0 of the IOADDR register.
3. *IOADDR.Configuration IO Access Enable* is cleared, to avoid un-intentional CSR read operations (that might cause a 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. CSR address is written to the IOADDR register where:
   a. Bit 31 (*IOADDR.Configuration IO Access Enable*) of the IOADDR register should be set to 1b.
   b. Bits 30:0 of IOADDR should hold the actual address of the internal register or memory being read.
2. CSR value is read from the IODATA register.
   a. The IODATA register is used as a window to the register or memory address specified by IOADDR register. As a result the data read from the IODATA register is the data of the CSR pointed to by bits 30:0 of the IOADDR register
3. *IOADDR.Configuration IO Access Enable* is cleared, to avoid un-intentional CSR read operations (that might cause a clear by read) by other applications scanning the configuration space.

Notes:

— In the event that the *CSR_conf_en* bit in the PCIe Init Configuration 2 NVM word is cleared, accesses to the IOADDR and IODATA registers via the configuration address space are ignored and have no effect on the register and the CSRs referenced by the *IOADDR* register. In this case, any read access to these registers returns a value of 0b.

— When Function is in D3 state Software should not attempt to access CSRs via the IOADDR and IODATA configuration registers.

— To enable CSR access via configuration space, Software should set bit 31 to 1b (*IOADDR.Configuration IO Access Enable*) of the IOADDR register. Software should clear bit 31 of the IOADDR register after completing CSR access to avoid an unintentional clear-by-read operation or by another application scanning the configuration address space.

— Bit 31 of the IOADDR register (*IOADDR.Configuration IO Access Enable*) has no effect when initiating access via I/O address space.

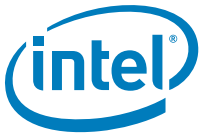
— Software should access CSRs via I/O address space or configuration address space but should not use both mechanisms at the same time.

8.1.2 Register Conventions

All registers in Foxville are defined to be 32 bits and should be accessed as 32-bit double-words; however, there are some exceptions to this rule:

— Register pairs where two 32-bit registers make up a larger logical size.

— Accesses to Flash memory (via Expansion ROM space, secondary BAR space, or the I/O space) might be byte, word or double word accesses.



Reserved bit positions: Some registers contain certain bits that are marked as reserved. These bits should never be set to a value of 1b by software. 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 specification 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 double words, partial writes (less than a 32- bit double word) do not take effect (the write is ignored). Partial reads returns all 32 bits of data regardless of the byte enables.

Note: Partial reads to Read-on-Clear (ICR) registers 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 bits statistics registers, the least significant 32-bit register should be read first.

Refer to the special notes for VLAN Filter Table, Multicast Table Arrays and Packet Buffer Memory, which appear in the specific register definitions.

Foxville register fields are assigned one of the attributes listed in Table 8-4.

Table 8-4. Foxville Register Field Attributes

Attribute	Description
RW	Read-Write field: Register bits are read-write and can be either set or cleared by software to the desired state.
RWM	Read-Write Modified field: Register bits are read-write and can be either set or cleared by software to the desired state. However, the value of this field might be modified by the hardware to reflect a status change.
RO	Read-only register: Register bits are read-only and should not be altered by software. Register bits might be initialized by hardware mechanisms such as pin straping, serial NVM or reflect a status of the hardware state.
ROM	Read-only Modified field: Register bits are read-only and will be either set or cleared by software upon read operation. However, the value of this field might be modified by the hardware to reflect a status change.
RW1C	Read-only status, Write-1-to-clear status register: Register bits indicate status when read, a set bit indicating a status event can be cleared by writing a 1b. Writing a 0b to RW1C bit has no effect.
Rsv	Reserved. Write 0b to these fields and ignore read.
RC	Read-only status, Read-to-clear status register: Register bits indicate status when read, a set bit indicating a status event is cleared by reading it.
SC	Self Clear field: a command field that is self clearing. These field are read as zero after the requested operation is done.
WO	Write only field: a command field that can not be read, These field read values are undefined.
RC/W	Read-Write status, Read-to-clear status register: Read-to-clear status register. Register bits indicate status when read. Register bits are read-write and can be either set or cleared by software to the desired state.

Table 8-4. Foxville Register Field Attributes

Attribute	Description
RC/W1C	Read-only status, Write-1-to-clear status register: Read-to-clear status register. Register bits indicate status when read, a set bit indicating a status event can be cleared by writing a 1b or by reading the register. Writing a 0b to RC/W1C bit has no effect.
RS	Read Set ‚Äì This is the attribute used for Semaphore bits. These bits are set by read in case the previous values were 0b. In this case the read value is 0b; otherwise the read value is 1b. Cleared by a write of 0b.
RW1	Read, Write-1 only register. Once a 1b has been written on a bit, the bit cannot be cleared to 0b.

PHY registers use a special nomenclature to define the read/write mode of individual bits in each register (see Table 8-5).

Table 8-5. PHY Register Nomenclature

Register Mode	Description
LH	Latched High. Event is latched and erased when read.
LL	Latched Low. Event is latched and erased when read. For example, Link Loss is latched when the PHY Control Register bit 2 = 0b. After read, if the link is good, the PHY Control Register bit 2 is set to 1b.
RO	Read Only.
RW	Read and Write.
SC	Self-Clear. The bit is set, automatically executed, and then reset to normal operation.
CR	Clear after Read.
Update	Value written to the register bit does not take effect until software PHY reset is executed.

Note: For all binary equations appearing in the register map, the symbol "|" is equivalent to a binary OR operation.

8.1.2.1 Registers Byte Ordering

This section defines the structure of registers that contain fields carried over the network. Some examples are L2, L3 and L4 fields.

The following example is used to describe byte ordering over the wire (hex notation):

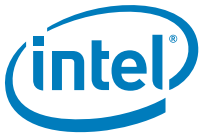
```
Last                          First
...,06, 05, 04, 03, 02, 01, 00
```

Each byte is sent with the LSbit first. That is, the bit order over the wire for this example is

```
Last                                        First
..., 0000 0011, 0000 0010, 0000 0001, 0000 0000
```

The general rule for register ordering is to use host ordering (also called little Endian). Using the previous example, a 6-byte fields (MAC address) is stored in a CSR in the following manner:

```
                  Byte 3  Byte 2  Byte 1  Byte0
DW address (N)    0x03    0x02    0x01    0x00
DW address (N+4)  ...     ...     0x05    0x04
```



The following exceptions use network ordering (also called big Endian). Using the previous example, a 16-bit field (EtherType) is stored in a CSR in the following manner:

```
                    Byte 3  Byte 2  Byte 1  Byte0
(DW aligned)        ...     ...     0x00    0x01
or
(Word aligned)      0x00    0x01    ...     ...
```

The following exceptions use network ordering:

All ETherType fields. For example, the *VET EXT* field in the *VET* register, the EType field in the *ETQF* register.

Note: The normal notation as it appears in text books, etc. is to use network ordering. For example, a MAC address of 00-A0-C9-00-00-00. The order on the network is 00, then A0, then C9, etc; however, the host ordering presentation would be:

```
                    Byte 3  Byte 2  Byte 1  Byte0
DW address (N)      00      C9      A0      00
DW address (N+4)    ...     ...     00      00
```

8.1.3 Register Summary

All Foxville's non-PCIe configuration registers, except for the MSI-X register, are listed in Table 8-6. These registers are ordered by grouping and are not necessarily listed in order that they appear in the address space.

Table 8-6. Register Summary

Offset		Abbreviation	Name	RW
General				
0x0000		CTRL	Device Control Register	RW
0x0008		STATUS	Device Status Register	RO
0x0018		CTRL_EXT	Extended Device Control Register	RW
0x0020		MDIC	MDI Control Register	RW
0x0028		FCAL	Flow Control Address Low	RW
0x002C		FCAH	Flow Control Address High	RW
0x0030		FCT	Flow Control Type	RW
0x0034		CONNSW	Copper/Fiber Switch Control	RW
0x0038		VET	VLAN Ether Type	RW
0x0170		FCTTV	Flow Control Transmit Timer Value	RW
0x0E00		LEDCTL	LED Control Register	RW
0x0E04		MDICNFG	MDC/MDIO Configuration Register	RW
0x1028		I2CCMD	SFP I^2C Command	RW
0x102C		I2CPARAMS	SFP I^2C Parameter	RW
0x1040		WDSTP	Watchdog Setup Register	RW
0x1044		WDSWSTS	Watchdog Software	RW
0x1048		FRTIMER	Free Running Timer	RWM

Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0x104C		TCPTimer	TCP Timer	RW
0x5B50		SWSM	Software Semaphore Register	RW
0x5B54		FWSM	Firmware Semaphore Register	RWM
0x5B5C		SW_FW_SYNC	Software-Firmware Synchronization	RWM
0x0E38		IPCNFG	Internal PHY Configuration	RW
0x0E14		PHPM	PHY Power Management	RW
NVM-Security Registers				
0x12010		EEC	EEPROM-Mode Control Register	RW
0x12014		EERD	EEPROM-Mode Read Register	RW
0x12018		EEWR	EEPROM-Mode Write Register	RW
0x1201C		FLA	Flash Access Register	RW
0x12114		FL_SECU	Flash Security Register	RW
0x12068		SHADOWINF	SHADOW RAM Information register	RO
0x12030		EEMNGCTL	Manageability EEPROM-Mode Control Register	RW
0x12108		FLFWUPDATE	Flash Firmware Code Update	RW
0x5B3C		VPDDIAG	VPD diagnostic register 0	RO
0x10208		VPDDIAG1	VPD diagnostic register 1	RO
0x12048		FLSWCTL	Software Flash Control Register	RW
0x1204C		FLSWDATA	Software Flash Burst Data Register	RW
0x12050		FLSWCNT	Software Flash Burst Access Counter	RW
0x1205C		FLERASEACK	Flash Erase Acknowledge Status Regitser	RC
Interrupts				
0x1500		ICR	Interrupt Cause Read	RC/W1C
0x1504		ICS	Interrupt Cause Set	WO
0x1508		IMS	Interrupt Mask Set/Read	RW
0x150C		IMC	Interrupt Mask Clear	WO
0x1510		IAM	Interrupt Acknowledge Auto Mask	RW
0x1520		EICS	Extended Interrupt Cause Set	WO
0x1524		EIMS	Extended Interrupt Mask Set/Read	RWM
0x1528		EIMC	Extended Interrupt Mask Clear	WO
0x152C		EIAC	Extended Interrupt Auto Clear	RW
0x1530		EIAM	Extended Interrupt Auto Mask	RW
0x1580		EICR	Extended Interrupt Cause Read	RC/W1C
0x1700 - 0x170C		IVAR	Interrupt Vector Allocation Registers	RW
0x1740		IVAR_MISC	Interrupt Vector Allocation Registers - MISC	RW
0x1680 - 0x16A0		EITR	Extended Interrupt Throttling Rate 0 - 4	RW
0x1514		GPIE	General Purpose Interrupt Enable	RW
0x5B68		PBACL	MSI-X PBA Clear	RW1C
0x5B88		PICAUSE	PCIe Interrupt Cause	RW1C
0x5B8C		PIENA	PCIe Interrupt Enable	RW
Receive				
0x0100		RCTL	Rx Control	RW

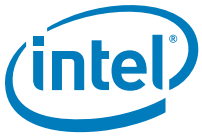


Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0x2160		FCRTL0	Flow Control Receive Threshold Low	RW
0x2168		FCRTH0	Flow Control Receive Threshold High	RW
0x2404		RXPBSIZE	Rx Packet Buffer Size	RW
0x2460		FCRTV	Flow Control Refresh Timer Value	RW
0xC000+0x40*n		RDBAL[0 - 3]	Rx Descriptor Base Low Queue	RW
0xC004+0x40*n		RDBAH[0 - 3]	Rx Descriptor Base High Queue	RW
0xC008+0x40*n		RDLEN[0 - 3]	Rx Descriptor Ring Length Queue	RW
0xC00C + 0x40 * n		SRRCTL[0 - 3]	Split and Replication Receive Control Register Queue	RW
0xC010+0x40*n		RDH[0 - 3]	Rx Descriptor Head Queue	RW
0xC018+0x40*n		RDT[0 - 3]	Rx Descriptor Tail Queue	RW
0xC028+0x40*n		RXDCTL[0 - 3]	Receive Descriptor Control Queue	RW
0xC014		RXCTL[0]	Receive Queue DCA CTRL Register	RW
0x5000		RXCSUM	Receive Checksum Control	RW
0x5004		RLPML	Receive Long packet maximal length	RW
0x5008		RFCTL	Receive Filter Control Register	RW
0x5200- 0x53FC		MTA[127:0]	Multicast Table Array (n)	RW
0x5400 + 8*n		RAL[0-15]	Receive Address Low (15:0)	RW
0x5404 + 8 *n		RAH[0-15]	Receive Address High (15:0)	RW
0x55B0		VLANPQF	VLAN Priority Queue Filter	RW
0x5480 – 0x549C		PSRTYPE[3:0]	Packet Split Receive type (n)	RW
0x5600-0x57FC		VFTA[127:0]	VLAN Filter Table Array (n)	RW
0x5818		MRQC	Multiple Receive Queues Command	RW
0x5C00-0x5C7C		RETA	Redirection Table	RW
0x5C80-0x5CA4		RSSRK	RSS Random Key Register	RW
Transmit Register Descriptions				
0x0400		TCTL	Tx Control	RW
0x0404		TCTL_EXT	Tx Control Extended	RW
0x0410		TIPG	Tx IPG	RW
0x041C		RETX_CTL	Retry Buffer Control	RW
0x3404		TXPBSIZE	Transmit Packet Buffer Size	RW
0x359C		DTXTCPFLGL	DMA Tx TCP Flags Control Low	RW
0x35A0		DTXTCPFLGH	DMA Tx TCP Flags Control High	RW
0x3540		DTXMXSZRQ	DMA Tx Max Total Allow Size Requests	RW
0x355C		DTXMXPKTSZ	DMA Tx Max Allowable Packet Size	RW
0x3590		DTXCTL	DMA Tx Control	RW
0xE000+0x40*n		TDBAL[0 - 3]	Tx Descriptor Base Low	RW
0xE004+0x40*n		TDBAH[0 - 3]	Tx Descriptor Base High	RW
0xE008+0x40*n		TDLEN[0 - 3]	Tx Descriptor Ring Length	RW
0xE010+0x40*n		TDH[0 - 3]	Tx Descriptor Head	RW
0xE018+0x40*n		TDT[0 - 3]	Tx Descriptor Tail	RW
0xE028+0x40*n		TXDCTL[0 - 3]	Transmit Descriptor Control Queue	RW

Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0xE014+0x40*n		TXCTL[0 - 3]	Tx DCA CTRL Register Queue	RW
0xE038+0x40*n		TDWBAL[0 - 3]	Transmit Descriptor WB Address Low Queue	RW
0xE03C+0x40*n		TDWBAH[0 - 3]	Transmit Descriptor WB Address High Queue	RW
Transmit Scheduling Registers				
0x300C + 0x40 *n		TQAVHC	Transmit Qav High Credits	RW
0x3570		TQAVCTRL	Transmit Qav Control	RW
0x3344 + 4*n		TXQCTL[n]	Transmit Queue Control	RW
0x3004, 0x3044		TQAVCC[n]	Tx Qav Credit Control	RW
0x3354		TxARB	Tx Arbitration Control	RW
0x3310		GTxOFFSET	Transmit Scheduling Offset	RW
0x3314		BASET_L	Scheduling Base Time Register Low	RW
0x3318		BASET_H	Scheduling Base Time Register High	RW
0x331C		QbvCycleT	Qbv Cycle Time	RW
0x3320		QbvCycleT_S	Shadow Qbv Cycle Time	RW
0x3324 + 4*n		StQT	Qbv Start Time	RW
0x3334 + 4*n		EndQT	Qbv End Time	RW
Filters				
0x5CB0 + 4*n		ETQF[0 - 7]	EType Queue Filter 0 - 7	RW
0x5A80 + 4*n		IMIR[0 - 7]	Immediate Interrupt Rx 0 - 7	RW
0x5AA0 + 4*n		IMIREXT[0 - 7]	Immediate Interrupt Rx Extended 0 - 7	RW
0x5AC0		IMIRVP	Immediate Interrupt Rx VLAN Priority	RW
0x55FC		SYNQF	SYN Packet Queue Filter	RW
Per Queue Statistics				
0x10010 + 0x100*n		PQGPRC[0 - 3]	Per Queue Good Packets Received Count	RW
0x10014 + 0x100*n		PQGPTC[0 - 3]	Per Queue Good Packets Transmitted Count	RW
0x10018 + 0x100*n		PQGORC[0 - 3]	Per Queue Good Octets Received Count	RW
0x10034 + 0x100*n		PQGOTC[0 - 3]	Per Queue Octets Transmitted Count	RW
0x10038 + 0x100*n		PQMPRC[0 - 3]	Per Queue Multicast Packets Received Count	RW
0xC030 + 0x40 * n		RQDPC[0 - 3]	Receive Queue Drop Packet Count Register 0 - 3	RW
0xE030 + 0x40 * n		TQDPC[0 - 3]	Transmit Queue Drop Packet Count Register 0 - 3	RW
Statistics				
0x4000		CRCERRS	CRC Error Count	RC
0x4004		ALGNERRC	Alignment Error Count	RC
0x400C		RXERRC	Rx Error Count	RC
0x4010		MPC	Missed Packets Count	RC
0x4014		SCC	Single Collision Count	RC
0x4018		ECOL	Excessive Collisions Count	RC

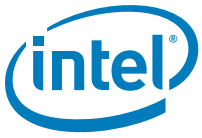


Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0x401C		MCC	Multiple Collision Count	RC
0x4020		LATECOL	Late Collisions Count	RC
0x4028		COLC	Collision Count	RC
0x402C		RERC	Receive Error Count	RC
0x4030		DC	Defer Count	RC
0x4034		TNCRS	Transmit - No CRS	RC
0x403C		HTDPMC	Host Transmit Discarded Packets by MAC Count	RC
0x4040		RLEC	Receive Length Error Count	RC
0x4048		XONRXC	XON Received Count	RC
0x404C		XONTXC	XON Transmitted Count	RC
0x4050		XOFFRXC	XOFF Received Count	RC
0x4054		XOFFTXC	XOFF Transmitted Count	RC
0x4058		FCRUC	FC Received Unsupported Count	RC
0x405C		PRC64	Packets Received (64 Bytes) Count	RC
0x4060		PRC127	Packets Received (65-127 Bytes) Count	RC
0x4064		PRC255	Packets Received (128-255 Bytes) Count	RC
0x4068		PRC511	Packets Received (256-511 Bytes) Count	RC
0x406C		PRC1023	Packets Received (512-1023 Bytes) Count	RC
0x4070		PRC1522	Packets Received (1024-1522 Bytes)	RC
0x4074		GPRC	Good Packets Received Count	RC
0x4078		BPRC	Broadcast Packets Received Count	RC
0x407C		MPRC	Multicast Packets Received Count	RC
0x4080		GPTC	Good Packets Transmitted Count	RC
0x4088		GORCL	Good Octets Received Count (Lo)	RC
0x408C		GORCH	Good Octets Received Count (Hi)	RC
0x4090		GOTCL	Good Octets Transmitted Count (Lo)	RC
0x4094		GOTCH	Good Octets Transmitted Count (Hi)	RC
0x40A0		RNBC	Receive No Buffers Count	RC
0x40A4		RUC	Receive Under Size Count	RC
0x40A8		RFC	Receive Fragment Count	RC
0x40AC		ROC	Receive Oversize Count	RC
0x40B0		RJC	Receive Jabber Count	RC
0x40B4		MNGPRC	Management Packets Receive Count	RC
0x40B8		MPDC	Management Packets Dropped Count	RC
0x40BC		MNGPTC	Management Packets Transmitted Count	RC
0x40C0		TORL	Total Octets Received (Lo)	RC
0x8FE0		B2OSPC	BMC2OS Packets Sent by MC	RC
0x4144		BMNGPTC	MC Management Packets Transmitted Count	RC
0x4158		B2OGPRC	BMC2OS Packets Received by Host	RC
0x8FE4		O2BGPTC	OS2BMC Packets Received by MC	RC
0x415C		O2BSPC	OS2BMC Packets Transmitted By Host	RC
0x40C4		TORH	Total Octets Received (Hi)	RC
0x40C8		TOTL	Total Octets Transmitted (Lo)	RC
0x40CC		TOTH	Total Octets Transmitted (Hi)	RC

Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0x40D0		TPR	Total Packets Received	RC
0x40D4		TPT	Total Packets Transmitted	RC
0x40D8		PTC64	Packets Transmitted (64 Bytes) Count	RC
0x40DC		PTC127	Packets Transmitted (65-127 Bytes) Count	RC
0x40E0		PTC255	Packets Transmitted (128-256 Bytes) Count	RC
0x40E4		PTC511	Packets Transmitted (256-511 Bytes) Count	RC
0x40E8		PTC1023	Packets Transmitted (512-1023 Bytes) Count	RC
0x40EC		PTC1522	Packets Transmitted (1024-1522 Bytes) Count	RC
0x40F0		MPTC	Multicast Packets Transmitted Count	RC
0x40F4		BPTC	Broadcast Packets Transmitted Count	RC
0x40F8		TSCTC	TCP Segmentation Context Transmitted Count	RC
0x4100		IAC	Interrupt Assertion Count	RC
0x4104		RPTHC	Rx Packets to Host Count	RC
0x4148		TLPIC	EEE Tx LPI Count	RC
0x414C		RLPIC	EEE Rx LPI Count	RC
0x4108		DBGC1	Debug counter 1	RC
0x410C		DBGC2	Debug counter 2	RC
0x4110		DBGC3	Debug counter 3	RC
0x411C		DBGC4	Debug counter 4	RC
0x4118		HGPTC	Host Good Packets Transmitted Count	RC
0x4120		RXDMTC	Rx Descriptor Minimum Threshold Count	RC
0x4128		HGORCL	Host Good Octets Received Count (Lo)	RC
0x412C		HGORCH	Host Good Octets Received Count (Hi)	RC
0x4130		HGOTCL	Host Good Octets Transmitted Count (Lo)	RC
0x4134		HGOTCH	Host Good Octets Transmitted Count (Hi)	RC
0x4138		LENERRS	Length Errors Count Register	RC
Wake Up and Proxying				
0x5800		WUC	Wake Up Control	RW
0x5808		WUFC	Wake Up Filter Control	RW
0x580C		WUFC_EXT	Wakeup Filter Control Register Extended	RW
0x5810		WUS	Wake Up Status Register	RW1C
0x5814		WUS_EXT	Wake Up Status Register Extended	RW1C
0x5F60		PROXYFC	Proxying Filter Control	RW
0x5F64		PROXYS	Proxying Status	RW1C
0x5838		IPAV	IP Address Valid	RW
0x5840- 0x5858		IP4AT	IPv4 Address Table	RW
0x5880- 0x588F		IP6AT	IPv6 Address Table	RW
0x5900		WUPL	Wake Up Packet Length	RO
0x5A00- 0x5A7C		WUPM	Wake Up Packet Memory	RO
0xB800 + 0x4*n		WUPM_EXT[n]	Wakeup Packet Memory Extended	RO
0x9000-0x93FC		FHFT	Flexible Host Filter Table Registers	RW
0x9A00-0x9DFC		FHFT_EXT	Flexible Host Filter Table Registers Extended	RW
0x5804		FHFTSL	Flex Filter indirect table select	RW

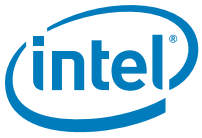


Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0x5590		PROXYFCEX	Proxy Filter Control Extended	RW
0x5594		PROXYEXS	Proxy Extended Status	RW1C
0x5500-0x557C		WFUTPF[31:0]	Wake Flex UDP TCP Port Filter	RW
0x5580		RFUTPF	Range Flex UDP TCP Port Filter	RW
0x5584		RWPFC	Range Wake Port Filter Control	RW
0x5588		WFUTPS	Wake Filter UDP TCP Status	RW1C
0x558C		WCS	Wake Control Status	RW1C
Management Register				
0x5820		MANC	Management Control	RW
0x5864		MNGONLY	Management Only Traffic Register	RW
Host Slave Interface				
0x8800-0x8EFC		HOST_INT_MEM	Host Interface Memory Address Space	RW
0x8F00		HICR	HOST Interface Control Register	RW
0x8F04		FWSWMB	Firmware Software Mailbox Register	RW
0x8F0C		FWSTS	Firmware Status Register	RW
0x8F10		SWSR	Software Status register	RW
0x8F50		MSG2MNG	Message to Manageability Register	RCW
0x8F54		MSKMSG2MNG	Message to Manageability Mask Register	RW
0x8F20		MNGPARCTL	Management Parity configuration Register	RW
0x8F24		MNGPARSTS	Management Parity status Register	RW1/C
0x8F28		MNGECCCTL	Management ECC configuration Register	RW
0x8F2C		MNGECCSTS	Management ECC status Register	RW1/C
0x8F40		HIBBA	Host Interface Buffer Base Address	RW
0x8F44		HIBMAXOFF	Host Interface Buffer Maximum Offset	RO
PCIe				
0x5B00		GCR	PCIe Control Register	RW
0x5B30		FACTPS	Function Active and Power State	RW
0x5B64		MREVID	Mirrored Revision ID	RO
0x5B6C		GCR_EXT	PCIe Control Extended Register	RW
				RW
0x12540		PTM_CTRL	PTM Control	RW
0x12544		PTM_STAT	PTM Status	RW1C
0x12548		PTM_STRT_TIME	PTM Cyclic Start Time	RW
0x1254C		PTM_CYCLE_CTRL	PTM Cycle Control	RW
0x12550		PCIe_DIG_Delay	PCIe Digital Latency	RW
0x12554		PCIe_PHY_Delay	PCIe PHY Latency	RO
0x12558		PTM_T1_TIM0_L	T1 on Timer 0 Low	RO
0x1255C		PTM_T1_TIM0_H	T1 on Timer 0 High	RO
0x12560		PTM_T1_TIM1_L	T1 on Timer 1 Low	RO
0x12564		PTM_T1_TIM1_H	T1 on Timer 1 High	RO
0x12568		PTM_T1_TIM2_L	T1 on Timer 2 Low	RO

Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0x1256C		PTM_T1_TIM2_H	T1 on Timer 2 High	RO
0x12570		PTM_T1_TIM3_L	T1 on Timer 3 Low	RO
0x12574		PTM_T1_TIM3_H	T1 on Timer 3 High	RO
0x12578		PTM_Prev_T4m1	T4 minus T1 on Previous PTM Cycle	RO
0x1257C		PTM_Curr_T4m1	T4 minus T1 on this PTM Cycle	RO
0x12580		PTM_Prev_T3m2	T3 minus T2 on Previous PTM Cycle	RO
0x12584		PTM_Prev_T2_L	T2 on Previous PTM Cycle Low	RO
0x12588		PTM_Prev_T2_H	T2 on Previous PTM Cycle high	RO
0x1258C		PTM_Curr_T2_L	T2 on this PTM Cycle Low	RO
0x12590		PTM_Curr_T2_H	T2 on this PTM Cycle High	RO
0x12594		PTM_TDELAY	PTM PCIe Link Delay	RO
0x125A4		PCIE_L12_TIMES	Time params for L12	RW
0x125A8		PCIE_TIMES_2P10	PCIE to P10 state	RW
0x125B0		PCIE_L1_EXTCLK	PCIE Ext CLK Warm up	RW
Memory Error Detection				
0x1084		PEIND	Parity and ECC Indication	RC
0x1088		PEINDM	Parity and ECC Indication Mask	RW
0x245C		PBECCSTS	Packet Buffer ECC Status	RW
0x5BA0		PCIEERRCTL	PCIe Parity Control Register	RW
0x5BA4		PCIEECCCTL	PCIe ECC Control Register	RW
0x5BA8		PCIEERRSTS	PCIe Parity Status Register	RW1C
0x5BAC		PCIEECCSTS	PCIe ECC Status Register	RW1C
0x5B7C		PCIEACL01	PCIe ACL0 and ACL1 Register	RW to FW
0x5B80		PCIEACL23	PCIe ACL2 and ACL3 Register	RW to FW
0x5F54		LANPERRCTL	LAN Port Parity Error Control Register	RW to FW
0x5F5C		LANPERRINJ	LAN Port Parity Error Inject Register	SC
0x5F58		LANPERRSTS	LAN Port Parity Error Status Register	RW1C
0x3F04		DRPARC	DMA Receive Parity and ECC Control	RW/SC
Power Management Registers				
0x5BB0		LTRMINV	Latency Tolerance Reporting (LTR) Minimum Values	RW
0x5BB4		LTRMAXV	Latency Tolerance Reporting (LTR) Maximum Values	RW
0x01A0		LTRC	Latency Tolerance Reporting (LTR) Control	RW
0x0E30		EEER	Energy Efficient Ethernet (EEE) Register	RW
0x0E34		EEE_SU	Energy Efficient Ethernet (EEE) Setup Register	RW
0x0E3C		EEE_SU_2P5	Energy Efficient Ethernet (EEE) 2.5Gb/s Setup Register	RW
Time Sync				
0xB620		TSYNCRXCTL	Rx Time Sync Control Register	RW
0xB614		TSYNCTXCTL	Tx Time Sync Control Register	RW
0xB678		DOM2TIMER	Rx Domain to 1588 Timer Mapping	RW
0xB618		TXSTMPL_0	Tx Timestamp Value Low 0	RO
0xB61C		TXSTMPH_0	Tx Timestamp Value High 0	RO
0xB698		TXSTMPL_1	Tx Timestamp Value Low 1	RO
0xB69C		TXSTMPH_1	Tx Timestamp Value High 1	RO

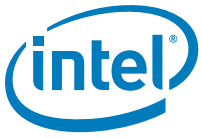


Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0xB6B8		TXSTMPL_2	Tx Timestamp Value Low 2	RO
0xB6BC		TXSTMPH_2	Tx Timestamp Value High 2	RO
0xB6D8		TXSTMPL_3	Tx Timestamp Value Low 3	RO
0xB6DC		TXSTMPH_3	Tx Timestamp Value High 3	RO
0xB6F8		SYSTIMR_0	System Time Residue Register 0	RW
0xB600		SYSTIML_0	System Time Register Low 0	RW
0xB604		SYSTIMH_0	System Time Register High 0	RW
0xB6FC		SYSTIMTM_0	System Time Register Tx MS 0	RW
0xB608		TIMINCA_0	Increment Attributes Register 0	RW
0xB60C		TIMADJ_0	Time Adjustment Offset Register 0	RW
0xB684		SYSTIMR_1	System Time Residue Register 1	RW
0xB688		SYSTIML_1	System Time Register Low 1	RW
0xB68C		SYSTIMH_1	System Time Register High 1	RW
0xB680		SYSTIMTM_1	System Time Register Tx MS 1	RW
0xB690		TIMINCA_1	Increment Attributes Register 1	RW
0xB694		TIMADJ_1	Time Adjustment Offset Register 1	RW
0xB6A4		SYSTIMR_2	System Time Residue Register 2	RW
0xB6A8		SYSTIML_2	System Time Register Low 2	RW
0xB6AC		SYSTIMH_2	System Time Register High 2	RW
0xB6A0		SYSTIMTM_2	System Time Register Tx MS 2	RW
0xB6B0		TIMINCA_2	Increment Attributes Register 2	RW
0xB6B4		TIMADJ_2	Time Adjustment Offset Register 2	RW
0xB6C4		SYSTIMR_3	System Time Residue Register 3	RW
0xB6C8		SYSTIML_3	System Time Register Low 3	RW
0xB6CC		SYSTIMH_3	System Time Register High 3	RW
0xB6C0		SYSTIMTM_3	System Time Register Tx MS 3	RW
0xB6D0		TIMINCA_3	Increment Attributes Register 3	RW
0xB6D4		TIMADJ_3	Time Adjustment Offset Register 3	RW
0xB640		TSAUXC	Auxiliary Control Register	RW
0xB644		TRGTTIML0	Target Time Register 0 Low	RW
0xB648		TRGTTIMH0	Target Time Register 0 High	RW
0xB64C		TRGTTIML1	Target Time Register 1 Low	RW
0xB650		TRGTTIMH1	Target Time Register 1 High	RW
0xB654		FREQOUT0	Frequency Out 0 Control Register	RW
0xB658		FREQOUT1	Frequency Out 1 Control Register	RW
0xB65C		AUXSTMPL0	Auxiliary Timestamp 0 Register Low	RW
0xB660		AUXSTMPH0	Auxiliary Timestamp 0 Register High	RO
0xB664		AUXSTMPL1	Auxiliary Timestamp 1 Register Low	RW
0xB668		AUXSTMPH1	Auxiliary Timestamp 1 Register High	RO
0x5F50		TSYNCRXCFG	Time Sync Rx Configuration	RW
0x003C		TSSDP	Time Sync SDP Configuration Register	RW
Time Sync Interrupt Registers				
0xB66C		TSICR	Time Sync Interrupt Cause Register	RC/W1C

Table 8-6. Register Summary (Continued)

Offset		Abbreviation	Name	RW
0xB674		TSIM	Time Sync Interrupt Mask Register	RW
Time Sync QAV Statistics				
0x4280		PRMPTDTCNT	Good TX Preempted Packets	RC
0x4298		PRMEVNTTCNT	TX Preemption event counter	RC
0x4284		PRMPTDRCNT	Good RX Preempted Packets	RC
0x429C		PRMEVNTRCNT	RX Preemption event counter	RC
0x4288		PRMPBLTCNT	Good TX Preemptable Packets	RC
0x428C		PRMPBLRCNT	Good RX Preemptable Packets	RC
0x4290		PRMEXPTCNT	Good TX Express packets	RC
0x4294		PRMEXPRCNT	Good RX Express packets	RC
0x42A0		PRMEXCPRCNT	QBV Exception Counter	RC

8.1.3.1 Alias Addresses

Certain registers maintain an alias address designed for backward compatibility with software written for the 82542. For these registers, the alias address is listed Table 8-6. Those registers can be accessed by software at either the new offset or the alias offset. It is recommended that software that is written solely for Foxville, use the new address offset.

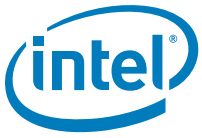
Note: For software written for both the 82542 and 82543, it might be preferable to use the alias addresses.

8.1.4 MSI-X BAR Register Summary

See description in Section 9.4.3.6.

Table 8-7. MSI-X Register Summary

Category	Offset	Abbreviation	Name	RW
MSI-X Table	0x0000 + n*0x10 [n=0...4]	MSIXTADD	MSI–X Table Entry Lower Address	RW
MSI-X Table	0x0004 + n*0x10 [n=0...4]	MSIXTUADD	MSI–X Table Entry Upper Address	RW
MSI-X Table	0x0008 + n*0x10 [n=0...4]	MSIXTMSG	MSI–X Table Entry Message	RW
MSI-X Table	0x000C + n*0x10 [n=0...4]	MSIXTVCTRL	MSI–X Table Entry Vector Control	RW
MSI-X Table	0x02000	MSIXPBA	MSIXPBA Bit Description	RO



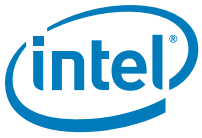
8.2 General Register Descriptions

8.2.1 Device Control Register - CTRL (0x00000; RW)

This register, as well as the Extended Device Control (CTRL_EXT) register, controls the major operational modes for the device. While software writes to this register to control device settings, several bits (such as FD and SPEED) can be overridden depending on other bit settings and the resultant link configuration determined by the PHY's auto-negotiation resolution. See Section 4.7.7 for details on the setup of these registers in the different link modes.

Field	Bit(s)	Initial Value	Description
FD	0	1b[1]	Full-Duplex Controls the MAC duplex setting when explicitly set by software. 0b = Half duplex. 1b = Full duplex.
Reserved	1	0b	Reserved Write 0b; ignore on read.
GIO Master Disable	2	0b	When set to 1b, the function of this bit blocks new master requests including manageability requests. If no master requests are pending by this function, the *STATUS.GIO Master Enable Status* bit is set. See Section 5.2.3.3 for further information.
Reserved	5:3	0x0	Reserved Write 0b, ignore on read.
SLU	6	0b[1]	Set Link Up Set Link Up must be set to 1b to permit the MAC to recognize the LINK signal from the PHY, which indicates the PHY has gotten the link up, and is ready to receive and transmit data. See Section 3.6.3 for more information about auto-negotiation and link configuration in the various modes. ***Notes:*** 1. The *CTRL.SLU* bit is normally initialized to 0b. However, if the *APM Enable* bit is set in the NVM then it is initialized to 1b. 2.
Reserved	7	0b[1]	3. Reserved '0'
Reserved	10:8	010b	Reserved
Reserved	11	0b[1]	Reserved
Reserved	12	0b	Reserved
Reserved	15:13	0x0	Reserved Write 0b, ignore on read.
SDP0_GPIEN	16	0b	General Purpose Interrupt Detection Enable for SDP0 If software-controlled I/O pin SDP0 is configured as an input, this bit (when 1b) enables the use for GPI interrupt detection.
SDP1_GPIEN	17	0b	General Purpose Interrupt Detection Enable for SDP1 If software-controlled I/O pin SDP1 is configured as an input, this bit (when 1b) enables the use for GPI interrupt detection.

Field	Bit(s)	Initial Value	Description
SDP0 DATA (RWM)	18	0b[1]	SDP0 Data Value Used to read or write the value of software-controlled I/O pin SDP0. If SDP0 is configured as an output (*SDP0_IODIR* = 1b), this bit controls the value driven on the pin (initial value NVM-configurable). If SDP0 is configured as an input, reads return the current value of the pin. When the SDP0_WDE bit is set, this field indicates the polarity of the watchdog indication. ***Note:*** The value of this field returns automatically to 0b whenever the selection of the internal signal to be driven to SDP0 pin is changed. E.g. when changing the signal from TimeSync to WoL.
SDP1 DATA (RWM)	19	0b[1]	SDP1 Data Value Used to read or write the value of software-controlled I/O pin SDP1. If SDP1 is configured as an output (SDP1_IODIR = 1b), this bit controls the value driven on the pin (initial value NVM-configurable). If SDP1 is configured as an input, reads return the current value of the pin. ***Note:*** The value of this field returns automatically to 0b whenever the selection of the internal signal to be driven to SDP0 pin is changed. E.g. when changing the signal from TimeSync to WoL.
ADVD3WUC	20	1b[1]	D3Cold Wake up Capability Enable When this bit is set to 0b, PME (WAKE#) is not generated in D3Cold. Bit loaded from NVM (refer to Section 6.1.1.32).
SDP0_WDE	21	0b[1]	SDP0 used for Watchdog Indication When set, SDP0 is used as a watchdog indication. When set, the *SDP0_DATA* bit indicates the polarity of the watchdog indication. In this mode, *SDP0_IODIR* must be set to an output.
SDP0_IODIR	22	0b[1]	SDP0 Pin Direction Controls whether software-controllable pin SDP0 is configured as an input or output (0b = input, 1b = output). Initial value is NVM-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
SDP1_IODIR	23	0b[1]	SDP1 Pin Direction Controls whether software-controllable pin SDP1 is configured as an input or output (0b = input, 1b = output). Initial value is NVM-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
Reserved	25:24	0x0	Reserved. Write 0b, ignore on read.
Reserved	26	0b	Reserved
RFCE	27	1b	Receive Flow Control Enable When set, indicates that Foxville responds to the reception of flow control packets. If auto-negotiation is enabled, this bit is set to the negotiated flow control value.
TFCE	28	0b	Transmit Flow Control Enable When set, indicates that Foxville transmits flow control packets (XON and XOFF frames) based on the receiver fullness. If auto-negotiation is enabled, this bit is set to the negotiated duplex value.



Field	Bit(s)	Initial Value	Description
DEV_RST (SC)	29	0b	Device Reset This bit performs a reset of the entire controller device, resulting in a state nearly approximating the state following a power-up reset or internal PCIe reset, except for system PCI configuration. 0b = Normal. 1b = Reset. This bit is self clearing. ***Notes:*** — Asserting *DEV_RST* generates an interrupt via the ICR.DRSTA interrupt bit. — Asserting *DEV_RST* sets the *STATUS.DEV_RST_SET* bit. — If the *PCIEERRCTL.PCIe Flush* bit is set to 1, device reset also clears PCIe transaction layer.
VME	30	0b	VLAN Mode Enable When set to 1b, VLAN information is stripped from all received 802.1Q packets. ***Note:*** If this bit is set, the *RCTL.SECRC* bit should also be set as the CRC is not valid anymore.
PHY_RST	31	0b	PHY Reset Generates a hardware-level reset to the internal PHY. 0b = Normal operation. 1b = Internal PHY reset asserted. Software should guarantee a minimum of 100usec at high for proper PHY reset assertion. De-asserting the PHY_RST, software should wait another 100usec for the completion of the PHY reset.

1. These bits are loaded from NVM.

8.2.2 Device Status Register - STATUS (0x0008; RO)

Field	Bit(s)	Initial Value	Description
FD	0	X	Full Duplex. 0b = Half duplex (HD). 1b = Full duplex (FD). Reflects duplex setting of the MAC and/or link. FD reflects the actual MAC duplex configuration. This normally reflects the duplex setting for the entire link, as it normally reflects the duplex configuration negotiated between the PHY and link partner (copper link) or MAC and link partner (fiber link).
LU	1	X	Link up. 0b = No link established. 1b = Link established. For this bit to be valid, the *Set Link Up* bit of the Device Control (CTRL.SLU) register must be set. ***Note:*** Link up provides a useful indication of whether something is attached to the port. Successful negotiation of features/link parameters results in link activity. The link start-up process (and consequently the duration for this activity after reset) can be several 100's of ms.
Reserved	3:2	X	Reserved Write 0b, ignore on read.
TXOFF	4	X	Transmission Paused This bit indicates the state of the transmit function when symmetrical flow control has been enabled and negotiated with the link partner. This bit is set to 1b when transmission is paused due to the reception of an XOFF frame. It is cleared (0b) upon expiration of the pause timer or the receipt of an XON frame.
Reserved	5	X	Reserved. Write 0b, ignore on read.

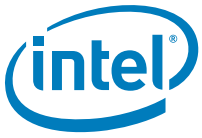
Field	Bit(s)	Initial Value	Description
SPEED	7:6	X	Link Speed Indication. This field together with the Speed_2P5 flag (bit 22) indicates the active link speed. When the Speed_2P5 flag is cleared, the SPEED field reports the following values: 00b = 10 Mb/s. 01b = 100 Mb/s. 10b = 1000 Mb/s. When the Speed_2P5 flag is set, the SPEED field equals to 10b.
RSVD	9:8	X	Reserved
PHYRA	10	1b	PHY Reset Asserted This read/write bit is set by hardware following the assertion of an internal PHY reset; it is cleared by writing a 0b to it. This bit is also used by firmware indicating a required initialization of the PHY.
Reserved	18:11	0x0	Reserved. Write 0b, ignore on read.
GIO Master Enable Status	19	1b	Cleared by Foxville when the *CTRL.GIO Master Disable* bit is set and no master requests are pending by this function and is set otherwise. Indicates that no master requests are issued by this function as long as the *CTRL.GIO Master Disable* bit is set.
DEV_RST_SET (RW1C)	20	0b	Device Reset Set When set, indicates that a device reset (*CTRL.DEV_RST*) was initiated by one of the software drivers. ***Note:*** Bit cleared by writing as 1b.
RST_DONE	21	1b	RST_DONE When set, indicates that software reset (*CTRL.DEV_RST*) has completed and the software device driver can begin initialization process.
Speed_2P5	22	0x0	Link Speed Indication for 2.5Gb/s. See the SPEED field in this register for a complete description.
Reserved	30:23	0x0	Reserved. Write 0b, ignore on read.
LPI_Ignore	30	0x0	Enable GTX clock out during LPI (ignore LPI for GTX clock). ***Note:*** When writing to this register the software should program this bit field to the same value as it was read. ***Note:*** Step C change: The functionality of this bit moves to EEER, bit 20
Reserved	31	0b[1]	Reserved

1. If the signature bits of the NVM's Initialization Control Word 1 match (01b), this bit is read from the NVM.

8.2.3 Extended Device Control Register - CTRL_EXT (0x0018; RW)

This register provides extended control of Foxville's functionality beyond that provided by the Device Control (CTRL) register.

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved. Write 0b, ignore on read.
Reserved	1	0b[1]	Reserved
SDP2_GPIEN	2	0b	General Purpose Interrupt Detection Enable for SDP2. If software-controllable I/O pin SDP2 is configured as an input, this bit (when set to 1b) enables use for GPI interrupt detection.
SDP3_GPIEN	3	0b	General Purpose Interrupt Detection Enable for SDP3. If software-controllable I/O pin SDP3 is configured as an input, this bit (when set to 1b) enables use for GPI interrupt detection.
Reserved	5:4	00b	Reserved. Write 0b, ignore on read.



Field	Bit(s)	Initial Value	Description
SDP2_DATA	6	0b[1]	SDP2 Data Value. Used to read (write) the value of software-controllable I/O pin SDP2. If SDP2 is configured as an output (SDP2_IODIR = 1b), this bit controls the value driven on the pin (initial value NVM-configurable). If SDP2 is configured as an input, reads return the current value of the pin.
SDP3_DATA	7	0b[1]	SDP3 Data Value. Used to read (write) the value of software-controllable I/O pin SDP3. If SDP3 is configured as an output (SDP3_IODIR = 1b), this bit controls the value driven on the pin (initial value NVM-configurable). If SDP3 is configured as an input, reads return the current value of the pin.
Reserved	9:8	0x0[1]	Reserved. Write 0b, ignore on read.
SDP2_IODIR	10	0b[1]	SDP2 Pin Direction. Controls whether software-controllable pin SDP2 is configured as an input or output (0b = input, 1b = output). Initial value is NVM-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
SDP3_IODIR	11	0b[1]	SDP3 Pin Direction. Controls whether software-controllable pin SDP3 is configured as an input or output (0b = input, 1b = output). Initial value is NVM-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
RSVD	12	0b	Reserved
EE_RST (SC)	13	0b	EEPROM Block Reset When set, initiates a reset-like event to the EEPROM block function. This causes an NVM auto-load operation as if a software reset had occurred. This bit is self-clearing.
Reserved	14	0x0	Reserved. Write 0b, ignore on read.
Reserved	15	0b	Reserved
NS_DIS	16	0	No Snoop Disable When set to 1b, Foxville does not set the no snoop attribute in any PCIe packet, independent of PCIe configuration and the setting of individual no snoop enable bits. When set to 0b, behavior of no snoop is determined by PCIe configuration and the setting of individual no snoop enable bits.
RO_DIS	17	0b	Relaxed Ordering Disabled When set to 1b, Foxville does not request any relaxed ordering transactions on the PCIe interface regardless of the state of bit 4 in the PCIe Device Control register. When this bit is cleared and bit 4 of the PCIe Device Control register is set, Foxville requests relaxed ordering transactions as specified by registers RXCTL and TXCTL (per queue and per flow).
Reserved	18	0b[1]	Reserved
Reserved	19	0b[1]	Reserved
PHY Power Down Ena	20	1b[1]	When set, the PHY enters a low-power state in the Dx power state and if the link is not requiredLoaded from bit 6 in the “Initialization Control Word 2” in the NVM.
Reserved	21	0b	Reserved. Write 0b, ignore on read.
Reserved	23:22	0x0[1]	Reserved
Reserved	24	0b	Reserved. Write 0b, ignore on read.
Reserved	25	0b[1]	Reserved
EXT_VLAN	26	0b[1]	External VLAN Enable When set, all incoming Rx packets are expected to have at least one VLAN with the Ether type as defined in *VET.EXT_VET* that should be ignored. The packets can have a second internal VLAN that should be used for all filtering purposes. All Tx packets are expected to have at least one VLAN added to them by the host. In the case of an additional VLAN request (*VLE* - VLAN Enable is set in transmit descriptor) the second VLAN is added after the first external VLAN is added by the host. This bit is reset only by a power up reset or by a full NVM auto-load and should only be changed while Tx and Rx processes are stopped.

Field	Bit(s)	Initial Value	Description
Reserved	27	0b	Reserved. Write 0b, ignore on read.
DRV_LOAD	28	0b	Driver Loaded This bit should be set by the software device driver after it is loaded. This bit should be cleared when the software device driver unloads or after a PCIe reset. The Management controller reads this bit to indicate to the manageability controller (BMC) that the driver has loaded. ***Note:*** Bit is reset on power-up or PCIe reset only.
Reserved	31:29	0b	Reserved Write 0b, Ignore on read.

1. These bits are read from the NVM.

Foxville enables up to four externally controlled interrupts. All software-definable pins, these can be mapped for use as GPI interrupt bits. Mappings are enabled by the *SDPx_GPIEN* bits only when these signals are also configured as inputs via *SDPx_IODIR*. When configured to function as external interrupt pins, a GPI interrupt is generated when the corresponding pin is sampled in an active-high state.

The bit mappings are listed in Table 8-8 for clarity.

Table 8-8. Mappings for SDI Pins Used as GPI

SDP Pin Used as GPI	CTRL_EXT Field Settings		Resulting ICR Bit (GPI)
	Direction	Enable as GPI Interrupt	
3	SDP3_IODIR	SDP3_GPIEN	14
2	SDP2_IODIR	SDP2_GPIEN	13
1	SDP1_IODIR	SDP1_GPIEN	12
0	SDP0_IODIR	SDP0_GPIEN	11

Note: If software uses the EE_RST function and desires to retain current configuration information, the contents of the control registers should be read and stored by software. Control register values are changed by a read of the NVM, which occurs after asserting the EE_RST bit.

Note: The NVM reset function can read configuration information out of the NVM, which affects the configuration of PCIe space BAR settings. The changes to the BARs are not visible unless the system reboots and the BIOS is allowed to re-map them.

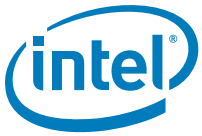
The *SPD_BYPS* bit performs a similar function to the *CTRL.FRCSPD* bit in that Foxville's speed settings are determined by the value software writes to the *CRTL.SPEED* bits. However, with the *SPD_BYPS* bit asserted, the settings in *CTRL.SPEED* take effect immediately rather than waiting until after Foxville's clock switching circuitry performs the change.

8.2.4 Media Dependent Interface (MDI) Control Register - MDIC (0x0020; RW)

Software uses this register to read or write MDI registers in the internal PHY or an external SGMII PHY.

Refer to Section 3.6.2.2.1 for details on usage of this register. The PHY registers described in Section 8.26 are accessible using the MDIC register.

Note: Register reset only by LAN_PWR_GOOD.



Field	Bit(s)	Initial Value	Description
DATA	15:0	X	Data In a Write command, software places the data bits and the MAC shifts them out to the PHY. In a Read command, the MAC reads these bits serially from the PHY and software can read them from this location.
REGADD	20:16	0x0	PHY Register Address: Reg. 0, 1, 2,...31
Reserved	25:21	0x0	Reserved. Write 0b, ignore on read. (Formally PHYADD field should be kept reserved to support legacy drivers).
OP	27:26	0x0	Opcode 01b = MDI write. 10b = MDI read. All other values are reserved.
R (RWM)	28	1b	Ready Bit Set to 1b by Foxville at the end of the MDI transaction (for example, indication of a read or write completion). It should be reset to 0b by software at the same time the command is written.
MDI_IE	29	0b	Interrupt Enable When set to 1b an Interrupt(*ICR.MDAC*) is generated at the end of an MDI cycle to indicate an end of a read or write operation to the PHY.
MDI_ERR (RWM)	30	0b	Error This bit is set to 1b by hardware when it fails to complete an MDI read. Software should make sure this bit is clear (0b) before issuing an MDI read or write command. ***Note:*** This bit is valid only when the *Ready* bit is set.
Reserved	31	0b	Reserved. Write 0b, ignore on read. (Formally Destination bit should be kept reserved to support legacy drivers).

8.2.5 LED Control - LEDCTL (0x0E00; RW)

This register controls the setup of the LEDs. Refer to Section 3.4.3 for details of the *Mode* fields encoding.

Note: Register reset only on LAN_PWR_GOOD.

Field	Bit(s)	Initial Value	Description
LED0_MODE	3:0	0110b[1]	LED0/LINK# Mode This field specifies the control source for the LED0 output. An initial value of 0110b selects the LINK100# indication.
LED_PCI_MODE	4	0b	0b = Use LEDs as defined in the other fields of this register. 1b = Use LEDs to indicate PCI3 lanes idle status in SDP mode (only when the led_mode is set to 0x8 – SDP mode) LED0 indicates electrical idle status.
GLOBAL_BLINK_MODE	5	0b[1]	Global Blink Mode This field specifies the blink mode of all the LEDs. 0b = Blink at 200 ms on and 200 ms off. 1b = Blink at 83 ms on and 83 ms off.

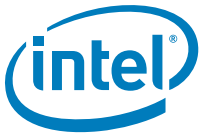
Field	Bit(s)	Initial Value	Description
LED0_IVRT	6	0b[1]	LED0/LINK# Invert This field specifies the polarity / inversion of the LED source prior to output or blink control. 0b = Do not invert LED source (LED active low). 1b = Invert LED source (LED active high). ***Note:*** In mode 0100b (link/activity) this field must be 0. The LED signal must be active low in mode 0100b (link/activity).
LED0_BLINK	7	0b[1]	LED0/LINK# Blink This field specifies whether to apply blink logic to the (possibly inverted) LED control source prior to the LED output. 0b = Do not blink asserted LED output. 1b = Blink asserted LED output.
LED1_MODE	11:8	0100b[1]	LED1/LINK/ACTIVITY This field specifies the control source for the LED1 output. An initial value of 0100b selects the LINK/ACTIVITY indication. When asserted, means the LINK indication and when BLINK means LINK and ACTIVITY.
Reserved	12	0b	Reserved Write as 0,ignore on read.
Reserved	13	0b	Reserved
LED1_IVRT	14	0b[1]	LED1/ACTIVITY# Invert This field specifies the polarity / inversion of the LED source prior to output or blink control. 0b = Do not invert LED source (LED active low). 1b = Invert LED source (LED active high). ***Note:*** In mode 0100b (link/activity) this field must be 0. The LED signal must be active low in mode 0100b (link/activity).
LED1_BLINK	15	1b[1]	LED1/ACTIVITY# Blink
LED2_MODE	19:16	0111b[1]	LED2/LINK1000# Mode This field specifies the control source for the LED2 output. An initial value of 0111b selects the LINK1000# indication.
Reserved	21:20	0x0	Reserved. Write 0b, ignore on read.
LED2_IVRT	22	0b[1]	LED2/LINK100# Invert This field specifies the polarity / inversion of the LED source prior to output or blink control. 0b = Do not invert LED source (LED active low). 1b = Invert LED source (LED active high). ***Note:*** In mode 0100b (link/activity) this field must be 0. The LED signal must be active low in mode 0100b (link/activity).
LED2_BLINK	23	0b[1]	LED2/LINK100# Blink
Reserved	27:24	0000b	Reserved.
Reserved	29:28	0x0	Reserved. Write 0x0, ignore on read.
Reserved	31:30	00b	Reserved.

1. These bits are read from the NVM.

8.2.6 MDC/MDIO Configuration Register - MDICNFG (0x0E04; RW)

This register is used to configure the MDIO connection that is accessed via the MDIC register. Refer to Section 3.6.2.2.1 for details on usage of this register.

Note: Register reset only by LAN_PWR_GOOD.



Field	Bit(s)	Initial Value	Description
Reserved	20:0	0x0	Reserved. Write 0b, ignore on read.
PHYADD	25:21	0x00	Determines the PHY ADDR that can be set to: 0x0, 0x2 or 0x1F. It is loaded from the Initialization Control 4 NVM word.
CLAUSE_45_en	26	0x0	Clause 45 Enable. 1b = clause 45 transactions 0b = clause 22 transactions This flag is write only while reading it does not reflects the device settings.
Reserved	30:25	0x0	Reserved. Write 0b, ignore on read.
Reserved	31	0b	Reserved

8.2.7 Copper/Fiber Switch Control - CONNSW (0x0034; RW)

Field	Bit(s)	Initial Value	Description
Reserved	1:0	00b	Reserved
Reserved	2	0b	***Note:*** Reserved '0'
Reserved	8:3	0x0	Reserved. Write 0x0, ignore on read.
Reserved	9	X	Reserved
PHYSD (RO)	10	X	PHY Signal Detect Indication Valid only if the receiver is not in electrical idle.
PHY_PDN (RO)	11	X	This bit indicates that the internal GbE PHY is in power down state. 0b = Internal GbE PHY not in power down. 1b = Internal GbE PHY in power down.
Reserved	31:12	0x0	Reserved. Write 0x0, ignore on read.

8.2.8 VLAN Ether Type - VET (0x0038; RW)

This register is used by hardware to identify 802.1Q (VLAN) Ethernet packets by comparing the Ether Type field carried by packets with the field contents. To be compliant with the 802.3ac standard, the *VET.VET* field has a value of 0x8100.

Field	Bit(s)	Initial Value	Description
VET (RO)	15:0	0x8100	VLAN EtherType. The MS byte in this field is the MS byte of the Ethertype which is first on the wire.
VET EXT	31:16	0x8100	External VLAN Ether Type. The MS byte in this field is the MS byte of the Ethertype which is first on the wire.

8.3 Internal Packet Buffer Size Registers

The following registers define the size of the on-chip receive and transmit buffers used to receive and transmit packets. Programming these registers automatically initializes internal packet-buffer RAM pointers. Refer to Section 4.7.9 for the general setting rule that applies on all these packet buffers.

The registers in this chapter reset only on power up.

8.3.1 RX Packet Buffer Size - RXPBSIZE (0x2404; RW)

Field	Bit(s)	Init.	Description
RXPBSIZE_EXP	5:0	0x22	High Priority receive packet buffer size in KB units. When preemption is enabled, it is used for the Express traffic. The high + low priority sizes should be less equal to 34 KB.
Bmc2ospbsize	11:6	0x02	BMC to OS packet buffer size in KB.
RXPBSIZE_BE	17:12	0x0	Low Priority receive packet buffer size in KB units. When preemption is enabled, it is used for the BE traffic. The high + low priority sizes should be less equal to 34 KB.
Reserved	30:18	0x0	Reserved. Write 0b, ignore on read.
cfg_ts_en	31	0x0	If set, a line is saved (16 bytes) per packet in the Rx packet buffer for the timestamp descriptor. If not set, no timestamp in packet support.

8.3.2 Transmit Packet Buffer Size - TXPBSIZE (0x3404; RW)

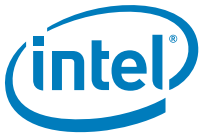
Field	Bit(s)	Init,	Description
Txpb0size	5:0	0x14	Tx Packet Buffer 0 Size in KB. In Qav mode, it controls the size in KB of the TXPB0, which is associated to TxQ0.
Txpb1size	11:6	0x0	In Qav mode, it controls the size in KB of the TXPB1, which is associated to TxQ1.
Txpb2size	17:12	0x0	In Qav mode, it controls the size in KB of the TXPB2, which is associated to TxQ2.
Txpb3size	23:18	0x0	In Qav mode, it controls the size in KB of the TXPB3, which is associated to TxQ3.
os2Bmcpbsize	29:24	0x4	OS to BMC packet buffer size in KB.
Reserved	31:30	0x0	Reserved Write 0b, ignore on read.

8.4 NVM Registers Descriptions

8.4.1 EEPROM-Mode Control Register - EEC (0x12010; RW)

This register provides software direct access to the NVM.

Field	Bit(s)	Init.	Description
Reserved	5:0	0x0	Reserved. Reads as 0b.
FLASH_IN_USE (RO)	6	0b	Valid when ee_pres = 1b . When this bit is set to 1b, it indicates that the NVM is present with a valid signature and the hardware was programmed from the NVM. 0b = NVM is not used. 1b = NVM is used.



Field	Bit(s)	Init.	Description
Reserved	7	0b	Reserved. Ignore on read.
EE_PRES (RO)	8	1b	Valid when auto_rd=1. When this bit is set, it indicates that a Flash is present and has the correct signature field, and the shadow RAM contains the auto-load information from one of those sources (no need for software programming).
Auto_RD (RO)	9	0b	NVM Auto-Read Done. When set to 1b, this bit indicates that the auto-read by hardware from the NVM is done. This bit is also set when the NVM is not found or when its signature field is not valid. This bit doesn't include MNG auto-load status.
Reserved	10	0b	Reserved.
EE_Size (RO)	14:11	0101b	NVM Size via EEPROM-Mode. This field defines the size of the NVM that is accessible via EEPROM-mode. This is equal to the size of the internal shadow RAM, fixed to 4 KB. It is encoded in power of 2 defined in K bit units.
PCI_ANA_done (RO)	18:15	0b	Reserved.
FLASH_DETECTED (RO)	19	0b	RO; Flash responded as not busy to a read status and returned a manufacturer ID.
Reserved	21:20	0x0	Reserved. Reads as 0x0.
Shadow_modified (RO)	22	0b	Indicates that shadow was modified.
FLUPD	23	0b	Flash Update. Writing 1b to this bit causes the content of the internal 4 KB shadow RAM to be written into one of the first two 4 KB sectors of the Flash device (Sector 0 or Sector 1). The bit is self-cleared immediately. When the bit is set, hardware issues an interrupt to firmware. No dump performed if FLASH_IN_USE bit is 0b.
Reserved	24	0b	Reserved. Reads as 0b.
SEC1VAL (RO)	25	0b	Sector 1 Valid. When set to 1b, indicates that the content of the 4 KB Sector 1 (from byte address 0x1000 to 0x1FFF) of the Flash device is valid. When set to 0b, indicates that the content of Sector 0 (from byte address 0x0000 to 0x0FFF) is valid. Meaningful only when EE_PRES bit and FLASH_IN-USE bit are read as 1b. This bit is RW to firmware.
FLUDONE (RO)	26	0b	Flash Update Done. When set to 1b, indicates that the Flash update process that was initiated by setting FLUPD bit has completed. This bit is cleared by hardware when FLUPD is set by host, and it is then set by firmware. Firmware shall also set this bit at init time. When FLUPD is set to 1b by FW (debug mode), FW shall write this bit with 0b for clearing it.
Reserved	31:27	0x0	Reserved. Reads as 0x0.

8.4.2 EEPROM-Mode Read Register - EERD (0x12014; RW)

This register is used by software to read individual words from the internal shadow RAM that reflects the first valid 4 KB sector of the NVM. To read a word, software writes the address to the *Read Address* field. Writing the register sets the Done bit to 0b. Foxville then reads the word from the internal shadow RAM and places it in the *Read Data* field, setting the *Read Done* field to 1b. Software can poll this register, looking for a 1b in the *Read Done* field and then using the value in the *Read Data* field.

Field	Bit(s)	Init.	Description
CMDV (RO)	0	0b	Command Valid Bit. This bit is cleared by hardware in case the read request was rejected.
DONE (RO field)	1	1b	Read Done. Set this bit to 1b when the EEPROM-mode read completes. Set this bit to 0b when the EEPROM-mode read is in progress. Note that writes by software are ignored.

Field	Bit(s)	Init.	Description
ADDR	12:2	0x0	Read Address. This field is written by software to indicate the address of the word to read.
Reserved	15:13	0x0	Reserved. Reads as 0x0.
DATA (RO field)	31:16	0x0	Read Data. Data returned from the EEPROM-Mode read.

8.4.3 EEPROM-Mode Write Register – EEWR (0x12018; RW)

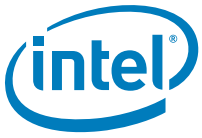
This register is used by software to write individual words in the internal shadow RAM that is about to reflect the first valid 4 KB sector of the NVM. To write a word, software writes the address to the Write Address field and the data to the Write Data field. Foxville writes the word into the internal shadow RAM, setting the Write Done field to 1b. Software can poll this register, looking for a 1b in the Write Done field before the next write. The data is effectively copied into the Flash device by use of the EEC.FLUPD command.
When this register is used to write a word into the NVM, that word is not written to any of Foxville's internal registers even if it is normally a hardware-accessed word.

Field	Bit(s)	Init.	Description
CMDV	0	0b	Command Valid. This bit is cleared by hardware in case the write request was rejected.
DONE	1	1b	Write Done. Set this bit to 1b when the EEPROM-mode write completes. Set this bit to 0b when the EEPROM-mode write is in progress. Note that writes by software are ignored.
ADDR	12:2	0x0	Write Address. This field is written by software to indicate the address of the word to write.
RESERVED	15:13	0x0	Reserved. Reads as 0x0.
DATA	31:16	0x0	Write Data. Data to be written into the shadow RAM.

8.4.4 Flash Access - FLA (0x1201C; RW)

This register provides software direct access to the Flash. Software can control the Flash by successive writes to this register. Data and address information is clocked into the NVM by software toggling the FL_SCK in this register. Data output from the Flash is latched into bit 3 of this register via the internal 125 MHz clock and can be accessed by software via reads of this register.

Field	Bit(s)	Init.	Description
FL_SCK	0	0b[2]	Clock Input to 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. This bit is not operational by the host when in the NVM Secure mode.
FL_CS	1	1b[2]	Chip Select Input to Flash. When FL_GNT is set to 1b, the FL_CS output signal is mapped to the chip select of the Flash device. Software enables the Flash by writing a 0b to this bit. This bit is not operational by the host when in the NVM Secure mode.



Field	Bit(s)	Init.	Description
FL_SI	2	0b[2]	Data Input to 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. This bit is not operational by the host when in the NVM Secure mode.
FL_SO (RO)	3	0b	Data Output Bit From Flash. The FL_SO input 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. RO bit.
FL_REQ	4	0b[2]	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. This bit is not operational by the host when in the NVM Secure mode.
FL_GNT (RO)	5	0b	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.
LOCKED (RO)	6	0b	A bit indicating (when set to 1b) that the NVM is in Secure mode. When set to 0b, the NVM is in Non-secure mode.
FLA_ABORT (RO)	7	0b[2]	Bit is RO, set by hardware when Flash access was aborted due to the deadlock avoidance. When this bit is set, further Flash bit banging access from this function is blocked. ***Note:*** This bit is cleared by a write of 1b to the FLA.FLA_CLR_ERR bit.
FLA_CLR_ERR (SC)	8	0b	Clear Flash Access Error. A write of 1b to this bit clears the FLA.FLA_ABORT bit and enables further bit banging access to the Flash.
Reserved	9	0b	Reserved
Reserved	10	0b	Reserved.
Reserved	11	0b	Reserved
Reserved	13:12	00b	Reserved
Reserved	15:14	00b	Reserved
EIP (RO)	16	0b	Sector Erase In Progress. Indicates that the Flash is in a sector erase cycle. RO bit.
FL_SIZE (RO)	19:17	000b[1]	Flash Size. Indicates the size of the Flash device according to the following equation: Size = 64 KB * 2 ** "FL_SIZE". The Flash size limits the range host memory mapped flash accesses and of Expansion ROM BAR mapped accesses to the Expansion ROM module beginning up to the Flash device's end. Supported Flash sizes: 000b = 0 - no valid Flash contents or no Flash device. 011b = 0.5 MB. 100b = 1 MB. 101b = 2 MB. 110b = 4 MB. 111b = 8 MB. This field is written by hardware from NVM words 0x11 after LAN_POWER_GOOD.
Reserved	27:20	0x0	Reserved.
BAR_ERR (RC)	28	0[2]	Host memory BAR or Expansion ROM BAR flash access error. The bit is set by hardware in one of the following event: 1) No bytes are selected in the byte enabled 2) Previous request didn't start and new request arrived 3) Write attempt to RO words (or to the first two 4 KB sectors) when NVM security is enabled. 4) Out of range access when NVM security is enabled. ***Note:*** RC is only for host access, not for firmware
FLASH_BUSY (RO)	29	0b	This bit indicates that the Flash is busy processing a Flash transaction and should not be accessed.
FL_BAR_BUSY (RO)	30	0b	BAR write can be done only while this bit is set to 0b.
Reserved	31	0b	Reserved.

[1] These bits are read from the NVM.
[2] These bits also reset when PCIe resets.

8.4.5 Flash Security - FL_SECU (0x12114; RO to host, RW to FW)

Field	Bit(s)	Init.	Description
SECURITY_ENABLED (RO)	0	1	This bit reports the NVM Security Straping Option. When read as 0b, it means NVM is operated in the Non-Secure mode because of straping option settings.
BLOCK_ALL_SW_ACCESS	1	1	When set, write access to protected and valid NVM from host are blocked. The accesses are completed by hardware but are not performed into the flash or the shadow RAM. See Note 1 below
BLOCK_PROTECTED_SW_ACCESS	2	1	when set, write access to protected area in a protected and valid NVM from host are blocked. The accesses are completed by hardware but are not performed into the flash or the shadow RAM. See Note 1 below
Reserved	31:3	0	
Note 1: The NVM is open for programming regardless of the setting of this register if the Signature field in the "EEPROM Sizing and Protected Fields" word in the NVM is invalid.			

8.4.6 Shadow RAM Information Register - SHADOWINF (0x012068; RO)

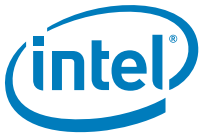
This register is a read-only register used for debugging.

Field	Bit(s)	Init.	Description
POWER_UP_STATUS	7:0	0xFF	Returns the status register value read from the flash device during power up.
JEDEC_ID	31:8	0xFFFFFF	Returns the manufacturer ID read during power up.

8.4.7 Manageability EEPROM-Mode Control Register – EEMNGCTL (0x12030; RO to Host, RW to FW)

Note: The transactions performed through this register are directed to/from the internal shadow RAM. The write data is effectively copied into the Flash device by use of the EEC. FLUPD command.

Field	Bit(s)	Initial value	Description
ADDR	10:0	0x0	Address. This field is written by manageability along with the WRITE bit to indicate which NVM word address to read or write. Only the first valid 4KB sector is accessible through this EEPROM-Mode interface dedicated to Manageability.
RESERVED	14:11	0x0	Reserved. Reads as 0b.
CMDV	15	0b	Command Valid. This bit is cleared by hardware in case the read/write request was rejected.



Field	Bit(s)	Initial value	Description
WRITE	16	0b	Write. This bit signals the device if the current operation is read or write. 0b = Read 1b = Write When using this register to write the CFG_DONE to 1b, this bit shall be written as 0b.
EEBUSY	17	0b	EEPROM-Mode Busy. This bit indicates that the internal shadow RAM is busy performing auto-load and should not be accessed.
CFG_DONE	18	0b	Manageability Configuration Cycle of the Port Completed. This bit indicates that the manageability configuration cycle (configuration of PHY) completed. It is cleared by hardware on PHY reset events, and it is set to 1b by firmware to indicate PHY configuration completed. Writing a 0b by firmware does not affect the state of this bit. When using this register to write the shadow RAM, this bit shall be written as 0b. Note: Software should not try to access the PHY for configuration before this bit is set.
RESERVED	30:19	0x0	Reserved.
DONE	31	1b	Transaction Done. This bit is cleared after the WRITE bit are set by manageability and is set back again when the shadow RAM write or read transaction completes.

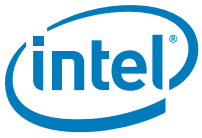
CMD{27:24}	Command Description	FLSWCNT.CNT range
0000b	Read	1 B - 4 KB
0001b	Write (must not cross a page of 256B boundary)	1 B - 256 B
0010b	Flash sector (4 KB) erase (when no security). The 4KB sector index to be erased is determined by the ADDR field.	X
0011b	Flash device erase (when no security). The entire Flash device is erased.	X
0100b	Read Status register of Flash device.	1 B - 4 KB
0101b	Write Status register of Flash device.	1 B - 4 KB
0110b	Write Enable. Depending on the Flash device Datasheet, this command might be needed prior to issuing the 1100b Programmable Write Status register opcode.	X
0111b	Reserved	X
1000b	Read JEDEC ID.	1 B - 4 KB
1001b	Reserved	X
1010b	Reserved	X
1011b	Programmable Read Status register (op-code 0x35 by default that can be re-programmed in FLASHGOP).	1 B - 4 KB
1100b	Programmable Write Status register (op-code 0x31 by default that can be re-programmed in FLASHGOP).	1 B - 4 KB

8.4.8 Flash Firmware Code Update – FLFWUPDATE (0x12108; RW)

Field	Bit(s)	Init.	Description
NEW_FW_ADD	11:0	0xFFF	Write by host only when auth done
RESERVED	28:12	0x0	Reserved.
AUTHEN_DONE	29	0b	Authentication Cycle Done. Set to 1b by firmware when done. This bit is self-cleared once the update request is set to 1b.
AUT_FAIL	30	0b	Authentication failed. Set to 1b by firmware when failed.
UPDATE	31	0b	Request authentication of the new secure section written. If the authentication succeeds, firmware resets itself to load its new code. This bit is self-cleared, always read as 0b.

8.4.9 Software Flash Control Register - FLSWCTL (0x12048; RW)

Field	Bit(s)	Initial value	Description
ADDR	23:0	0x0	Address in Bytes. This field is written by software along with CMD to indicate the Flash address to which the operation (read/write/erase, etc.) is performed. See the command description following this table.
CMD	27:24	0x0	Command. Indicates which command that should be executed. See table below for Flash Commands' list
CMDV	28	1b	Last Command was Valid. When cleared, it indicates that the last command issued was either a reserved combination (see the following table), or one of the following: ? When count reached zero (except for a general purpose status write) ? When a write burst crosses a Flash page ? When the address to be written is protected (RO) ? When the CNT specified is out of the permitted range (see the following table).
FLBUSY	29	0b	Flash Busy. This bit indicates that the Flash is busy processing a Flash transaction and should not be accessed.
DONE	30	1b	Single Flash Transaction Done. This bit clears after the register is written by software and is set back again when the single Flash transaction was issued to the Flash device. When writing a burst transaction, the bit is cleared every time software writes FLSWDATA.
GLDONE	31	1b	Global Flash Transaction Done. This bit clears after the register is written by software and is set back again when the all the Flash transactions were issued to the Flash device. For example, the Flash device completed all requested read/writes.



CMD{27:24}	Command Description	FLSWCNT.CNT range	Limitations to Host
0000b	Read	1 B - 4 KB	
0001b	Write (must not cross a page of 256B boundary)	1 B - 256 B	When in the NVM Secure mode, this command is operational only if applied on un-secured words.
0010b	Flash sector (4 KB) erase (when no security). The 4KB sector index to be erased is determined by the ADDR field.	X	When in NVM secure mode, this command is operational only if applied on un-secured sectors.
0011b	Flash device erase (when no security). The entire Flash device is erased.	X	This command is not operational when in NVM secure mode.
0100b	Read Status register of Flash device.	1 B - 4 KB	
0101b	Write Status register of Flash device.	1 B - 4 KB	This command is not operational when in NVM secure mode.
0110b	Write Enable. Depending on the Flash device Datasheet, this command might be needed prior to issuing the 1100b Programmable Write Status register opcode.	X	This command is not operational when in NVM secure mode.
0111b	Reserved	X	X
1000b	Read JEDEC ID.	1 B - 4 KB	X
1001b	Reserved	X	X
1010b	Reserved	X	X
1011b	Programmable Read Status register	1 B - 4 KB	Op-code cannot be reprogrammed when in NVM secured mode
1100b	Programmable Write Status register	1 B - 4 KB	This command is not operational when in NVM secure mode.

8.4.10 Software Flash Burst Data Register - FLSWDATA (0x1204C; RW)

Field	Bit(s)	Init.	Description
DATA	31:0	0x0	Burst NVM Data. Data written to or read from the NVM. When FLSWCNT.CNT field is programmed with a number of bytes that is not aligned a multiple of four (last Dword is a partial), the last valid byte(s) are located in the lower DATA field bytes.

8.4.11 Software Flash Burst Access Counter - FLSWCNT (0x12050; RW)

Restricted: This register should be extracted from external documentation.Note: This register can be read/write by manageability firmware and is read-only to host software.

Field	Bit(s)	Init.	Description
CNT	12:0	0x1	NVM Burst Counter. This counter holds the size in bytes of the Flash burst read or write.
RESERVED	31:13	0x0	Reserved.

8.4.12 VPD Diagnostic Register 0 - VPDDIAG (0x5B3C; RO to Host, RW to FW)

Field	Bit(s)	Init.	Description
VPD_EN	0	0	VPD Enabled. This bit is loaded from NVM word 0x0A.
VPD_ERR	7:1	0x0	VPD Error Code. 0x00 - No VPD error. 0x01 - No VPD read: NVM word 0x2F is 0xFFFF or VPD_EN bit is cleared in NVM word 0x0A. 0x02 - VPD area would overflow out of the shadow RAM size. 0x03 - First byte is not start tag (0x82). 0x04 - Wrong tags - an unfamiliar tag encountered. 0x05 - More than one RO/RW sections. 0x06 - VPD area used is longer than 1024. 0x07 - Internal error occurred. 0x08 - VPD area spreads across RW and RO areas of the shadow RAM **Notes**: - Errors 0x01-0x02 will NOT result in VPD_EN=0, but will rather cause silently drop of the VPD writes which are performed via VPD registers set. - Errors 0x03-0x06 will result in VPD made RO. - Error 0x07 indicates an internal error such as NVM read/write failure.
Reserved	31:8	0	

8.4.13 VPD Diagnostic Register 1 - VPDDIAG1 (0x10208; RO) DMA

Restricted: This register stores the VPD parameters as parsed by the auto-load process. This register is used for debug only and should be extracted from external documentation.

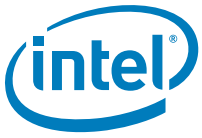
Field	Bit(s)	Init.	Description
VPD_ST_IS_BIG	0	0b	Set by hardware when VPD structure is bigger than 256 Bytes. In such a case, the VPD structure considered is 256 Bytes read only.
VPD_DONE	1	0b	Set when the VPD parsing process in NVM Auto-Read process is done.
VPD_RW	16:2	0x0	NVM address in words of the read write section start, within the VPD section in NVM.
VPD_RO	31:17	0x0	NVM address in words of the read only section start, within the VPD section in NVM.

8.5 Flow Control Register Descriptions

8.5.1 Flow Control Address Low - FCAL (0x0028; RW)

Flow control packets are defined by 802.3X to be either a unique multicast address or the station address with the EtherType field indicating PAUSE. The FCA registers provide the value hardware uses to compare incoming packets against, to determine that it should PAUSE its output.

The FCAL register contains the lower bits of the internal 48-bit Flow Control Ethernet address. All 32 bits are valid. Software can access the High and Low registers as a register pair if it can perform a 64-bit access to the PCIe bus. The complete flow control multicast address is: 0x01_80_C2_00_00_01; where 0x01 is the first byte on the wire, 0x80 is the second, etc.



Note: Any packet matching the contents of {*FCAH*, *FCAL*, *FCT*} when *CTRL.RFCE* is set is acted on by Foxville. Whether flow control packets are passed to the host (software) depends on the state of the *RCTL.DPF* bit.

Note: *FCAL* is reset only by PCIe reset and LAN_PWR_GOOD.

Field	Bit(s)	Initial Value	Description
FCAL	31:0	0x00C28001	Flow Control Address Low. The LS byte in this register is the MS byte of the MAC address which is first on the wire.

8.5.2 Flow Control Address High - FCAH (0x002C; RW)

This register contains the upper bits of the 48-bit Flow Control Ethernet address. Only the lower 16 bits of this register have meaning. The complete Flow Control address is {*FCAH*, *FCAL*}.

The complete flow control multicast address is: 0x01_80_C2_00_00_01; where 0x01 is the first byte on the wire, 0x80 is the second, etc.

Note: *FCAH* is reset only by PCIe reset and LAN_PWR_GOOD.

Field	Bit(s)	Initial Value	Description
FCAH	15:0	0x0100	Flow Control Address High. Should be programmed with 0x01_00. The MS byte in this field is the LS byte of the MAC address which is last on the wire.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.5.3 Flow Control Type - FCT (0x0030; RW)

This register contains the *Type* field that hardware matches to recognize a flow control packet. Only the lower 16 bits of this register have meaning. This register should be programmed with 0x88_08. The upper byte is first on the wire *FCT*[15:8].

Note: *FCT* is reset only by PCIe reset and LAN_PWR_GOOD.

Field	Bit(s)	Initial Value	Description
FCT	15:0	0x8808	Flow Control Type.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.5.4 Flow Control Transmit Timer Value - FCTTV (0x0170; RW)

The 16-bit value in the *TTV* field is inserted into a transmitted frame (either XOFF frames or any PAUSE frame value in any software transmitted packets). It counts in units of slot time of 64 bytes. If software needs to send an XON frame, it must set *TTV* to 0x0 prior to initiating the PAUSE frame.

Field	Bit(s)	Initial Value	Description
TTV	15:0	X	Transmit Timer Value.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.5.5 Flow Control Receive Threshold Low - FCRTL0 (0x2160; RW)

This register contains the receive threshold used to determine when to send an XON packet The complete register reflects the threshold in units of bytes. The lower four bits must be programmed to 0x0 (16 byte granularity). Software must set *XONE* to enable the transmission of XON frames. Each time hardware crosses the receive-high threshold (becoming more full), and then crosses the receive-low threshold and *XONE* is enabled (1b), hardware transmits an XON frame. When XONE is set, the *RTL* field should be programmed to at least 0x3 (at least 48 bytes).

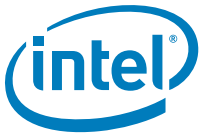
Flow control reception/transmission are negotiated capabilities by the auto-negotiation process. When Foxville is manually configured, flow control operation is determined by the *CTRL.RFCE* and *CTRL.TFCE* bits.

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0x0	Reserved. Write 0x0, ignore on read.
RTL0	16:4	0x0	Receive Threshold Low. FIFO low water mark for flow control transmission when Transmit Flow control is enabled (CTRL.TFCE = 1b). An XON packet is sent if the occupied space in the packet buffer is smaller or equal than this watermark. This field is in 16 bytes granularity. In legacy mode (one Rx packet buffer) this field is relevant for the whole Rx packet buffer. In TSN mode / VLAN priority mode it is relevant for the Express packet buffer / high Priority VLAN packet buffer respectively.
RTL1	29:17	0x0	Same functionality as the RTL0 for the Preemption packet buffer / Low Priority VLAN packet buffer in TSN mode / VLAN priority mode respectively. This field is NOT relevant in legacy mode and must be set to zero.
Reserved	30	0x0	Reserved. Write 0x0, ignore on read.
XONE	31	0b	XON Enable. 0b = Disabled. 1b = Enabled.

8.5.6 Flow Control Receive Threshold High - FCRTH0 (0x2168; RW)

This register contains the receive threshold used to determine when to send an XOFF packet. The complete register reflects the threshold in units of bytes. This value must be at maximum 48 bytes less than the maximum number of bytes allocated to the Receive Packet Buffer (*RXPBSIZE.RXPbsize*), and the lower four bits must be programmed to 0x0 (16 byte granularity). The value of *RTH* should also be bigger than *FCRTL0.RTL*. Each time the receive FIFO reaches the fullness indicated by *RTH*, hardware transmits a PAUSE frame if the transmission of flow control frames is enabled.

Flow control reception/transmission are negotiated capabilities by the auto-negotiation process. When Foxville is manually configured, flow control operation is determined by the *CTRL.RFCE* and *CTRL.TFCE* bits.



Field	Bit(s)	Initial Value	Description
Reserved	3:0	0x0	Reserved. Write 0x0, ignore on read.
RTH0	16:4	0x0	Receive Threshold High water mark for flow control transmission when Transmit Flow control is enabled (CTRL.TFCE = 1b). An XOFF packet is sent if the occupied space in the packet buffer is bigger or equal than this watermark. This field is in 16 bytes granularity. Refer to Section 3.6.4.3.1 to calculate recommended setting. When not in use must be program to 0. - The value programmed should be greater than maximum packet size. In legacy mode it is the whole Rx packet buffer. In TSN mode / VLAN mode it is relevant for Express packet buffer / higher priority packet buffer respectively.
RTH1	29:17	0x0	Same functionality as the RTH0 for the Preemption packet buffer / Low Priority VLAN packet buffer in TSN mode / VLAN priority mode respectively. This field is NOT relevant in legacy mode and must be set to zero.
Reserved	31:30	0x0	Reserved. Write 0x0, ignore on read.

8.5.7 Flow Control Refresh Threshold Value - FCRTV (0x2460; RW)

Field	Bit(s)	Initial Value	Description
FC_refresh_th	15:0	0x0	Flow Control Refresh Threshold. This value indicates the threshold value of the flow control shadow counter when transmit flow control is enabled (*CTRL.TFCE* = 1b). When the counter reaches this value, and the conditions for PAUSE state are still valid (buffer fullness above low threshold value), a PAUSE (XOFF) frame is sent to link partner. If this field contains zero value, the flow control refresh is disabled.
Reserved	31:16	X	Reserved. Write 0x0, ignore on read.

8.5.8 Flow Control Status - FCSTS0 (0x2464; RO)

This register describes the status of the flow control machine.

Field	Bit(s)	Initial Value	Description
Flow_control state	0	0b	Flow Control State Machine Signal. 0b = XON. 1b = XOFF.
Above high	1	0x0	The size of data in the memory is above the high threshold.
Below low	2	1b	The size of data in the memory is below the low threshold.
Reserved	15:3	0x0	Reserved. Write 0x0, ignore on read.
Refresh counter	31:16	0x0	Flow Control Refresh Counter.

8.6 PCIe Register Descriptions

8.6.1 PCIe Control - GCR (0x5B00; RW)

Note: Register resets by LAN_PWR_GOOD and PCIe reset.

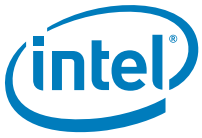
Field	Bit(s)	Initial Value	Description
Reserved	1:0	0x0	Reserved.
Discard on BME de-assertion	2	1b	When set and BME deasserted, PCIe discards all requests of this function.
Reserved	8:3	0x0	Reserved. Write 0x0, ignore on read.
Completion Timeout Resend Enable	9	0b[1]	When set, enables a resend request after the completion timeout expires. 0b = Do not resend request after completion timeout. 1b = Resend request after completion timeout. ***Note:*** This field is loaded from the *Completion Timeout Resend* bit in the NVM.
Reserved	10	0b	Reserved. Write 0b, ignore on read.
Number of Resends	12:11	11b	The number of resends in case of timeout or poisoned.
Reserved	17:13	0x0	Reserved. Write 0x0, ignore on read.
PCIe Capability Version (RO)	18	1b[2]	Reports the PCIe capability version supported. 0b = Capability version: 0x1. 1b = Capability version: 0x2.
Reserved	20:19	0x0	Reserved. Write 0, ignore on read.
Reserved	21	0b	Reserved
Reserved	28:22	0x0	Reserved. Write 0, ignore on read.
Reserved	29	0b	Reserved
Reserved	30	0b	Reserved Write 0, ignore on read.
DEV_RST In Progress	31	0b	Device Reset in Progress. This bit is set following device reset assertion (*CTRL.DEV_RST* = 1b) until no pending requests exist in PCIe. The software device driver should wait for this bit to be cleared before re-initializing the port (Refer to Section 4.3.1).

1. Loaded from *PCIe Completion Timeout Configuration* NVM word (word 0x15), reset on PCIe Power Good only.
2. The default value for this field is read from the *PCIe Init Configuration 3* NVM word (address 0x1A) bits 11:10. If these bits are set to 10b, then this field is set to 1b, otherwise field is reset to zero. Reset on PCIe Power Good only.

8.6.2 Function Active and Power State to MNG - FACTPS (0x5B30; RO)

Notes:

1. Register resets by LAN_PWR_GOOD and PCIe reset.
2. Firmware uses this register for configuration.



Field	Bit(s)	Initial Value	Description
Func Power State	1:0	00b	Power state indication of Function. 00b = DR. 01b = D0u. 10b = D0a. 11b = D3. This field resets only by LAN_PWR_GOOD.
Reserved	2	1b	Reserved.
Func Aux_En	3	0b	*Function Auxiliary (AUX) Power PM Enable* bit shadow from the configuration space.
Reserved	28:4	0x0	Reserved. Write 0x0, ignore on read.
MNGCG	29	0b	MNG Clock Gated. When set, indicates that the manageability clock is gated.
Reserved	30	0b	Reserved.
PM State Changed (RC)	31	0b	Indicates that one or more of the functional power states have changed. This bit is also a signal to the MC to create an interrupt. This bit is cleared on read by the MC. This bit resets only by LAN_PWR_GOOD.

8.6.3 Mirrored Revision ID - MREVID (0x5B64; RW)

Register resets by LAN_PWR_GOOD and PCIe reset.

Field	Bit(s)	Initial Value	Description
NVM RevID	7:0	0x0	Mirroring of revision ID loaded from the NVM in PCIe configuration space (from Device Rev ID word, address 0x1E).
Step REV ID	15:8	0x0 for A step 0x1 for B step	Revision ID from FUNC configuration space.
Reserved	31:16	0x0	Reserved Write 0x0, ignore on read.

8.6.4 PCIe Control Extended Register - GCR_EXT (0x5B6C; RW)

Register is reset by LAN_PWR_GOOD, PCIe reset, D3 to D0 move and FLR (Configuration reset).

Field	Bit(s)	Init.	Description
Reserved	3:0	0x0	Reserved.

Field	Bit(s)	Init.	Description
APBACD	4	0b	Auto PBA Clear Disable. When set to 1b, software can clear the PBA only by a direct write to clear access to the PBA bit. When set to 0b, any active PBA entry is cleared on the falling edge of the appropriate interrupt request to the PCIe block. The appropriate interrupt request is cleared when software sets the associated *Interrupt Mask* bit in the EIMS (re-enabling the interrupt) or by direct write to clear to the PBA.
Reserved	29:5	0x0	Reserved Write 0, ignore on read.
Reserved	30	0	Reserved

8.6.5 PTM Control - PTM_CTRL (0x12540; RW)

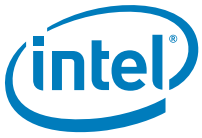
Reset by LAN_POWER_GOOD and PCIe Reset.

Field	Bit(s)	Initial value	Description
Reserved	1:0	0x0	Reserved
PTM_SHRT_CYC	7:2	0x2	The PTM_SHRT_CYC defined the time between two consecutive PTM requests when the data is not valid (defined in 2^10 nsec units =~1 usec). On top of it, it is guaranteed that the PTM request is not initiated sooner than 1usec after the reception of the previous PTM response.
PTM_TO	15:8	0x64	If a PTM respond message is not received within this time (defined in 2^10 nsec units =~1 usec), the collected PTM results are flushed and the PTM state machine starts from the beginning. This process is enabled for N_TO_MAX times max, and then will set the "retry error" bit. Note: Software should not set the PTM_TO to a lower value of 0x64 as required by the PCIe spec.
RSVD	~~30:16~~ 29:16	0x0	Reserved
Start_Now	29	0b	The Start_Now controls when the single PTM cycle or the cyclic PTM cycles will start. 0b - The PTM cycle starts at PTM_STRT_TIME 1b - The PTM cycle starts immediately ignoring the PTM_STRT_TIME
PTM_EN	30	0x0	Enable the PTM sequence on each triggered cycle. This option is enabled only if the PTM requester enable and PTM enable bits in the capability registers are set as well.
PTM_Trig	31	0x0	Setting this bit triggers the PTM cycle the same as it is initiated by the cyclic timer. Since this flag is NOT cleared autonomously, software should clear it before setting it again. Note: The PTM_Trig flag can be used only when the AUTO_CYC_EN flag in the PTM_CYCLE_CTRL register is cleared.

8.6.6 PTM Status - PTM_STAT (0x12544; RW1C)

Reset by LAN_POWER_GOOD and PCIe Reset.

Field	Bit(s)	Initial value	Description
PTM_DAT_VALID	0	0b	The PTM finished calculations and the registers are valid for read. To allow SW read a compatible set of registers, the PTM data valid is asserted every time a valid data is collected. The next PTM request will not be sent until the SW will reset (by writing 1) this bit.



Field	Bit(s)	Initial value	Description
PTM_RET_ERR	1	0b	This flag is set after an un-responsive requests from the root port (time out)
Bad_PTM_RES	2	0b	PTM Response message is received instead of a PTM Response Data.
T4mT1_OVFL	3	0b	T4 minus T1 overflow.
ADJUST_1ST	4	0b	The 1588 timer was adjusted during the first PTM cycle
ADJUST_CYC	5	0b	The 1588 timer was adjusted during non-first PTM cycle
RSVD	13:6	0x0	Reserved
RSVD	15:14	0x0	Reserved for Endianess indication by the FW in the "PTM Message". The encoding of this field in the "PTM Message" is shown below: 00b - Little Endian notation 01b - Big Endian notation 10b - "Ambiguous Endianess" error indication 11b - Reserved
RSVD	31:16	0x0	Reserved

8.6.7 PTM Start Time - PTM_STRT_TIME (0x12548; RW)

Reset by LAN_POWER_GOOD and PCIe Reset.

Field	Bit(s)	Initial value	Description
STRT_TIME	31:0	0x0	The sub second time on which the PTM sequences should start. Using the Start_Time option, the Start_Now flag in the PTM_CTRL register must be cleared.

8.6.8 PTM Cycle Control - PTM_CYCLE_CTRL (0x1254C; RW)

Reset by LAN_POWER_GOOD and PCIe Reset.

Field	Bit(s)	Initial value	Description
PTM_CYC_TIME	9:0	0x0	Time between consecutive PTM requests in 2^20 nsec units (~1 msec units).
RSVD	27:10	0x0	Reserved
PTM_CYC_SEL	29:28	0x0	Selects the 1588 timer the Host timer is compared with.
Reserved	30	0x0	Reserved
AUTO_CYC_EN	31	0x0	Enable the cyclic triggering of the sync cycles. Note that programming this flag initializes the Cyclic PTM state machine.

8.6.9 PCIe Digital Delay - PCIe_DIG_Delay (0x12550; RW)

Reset by LAN_POWER_GOOD, PCIe Reset and software reset

Field	Bit(s)	Initial value	Description
DIG1_TX_LATE	7:0	0x2B	Tx Digital delay on PCIe Gen 1 in nsec units (43nsec). Software should program this register to 0x0.
DIG2_TX_LATE	15:8	0x17	Tx Digital delay on PCIe Gen 2 in nsec units (23nsec). Software should program this register to 0x0.

Field	Bit(s)	Initial value	Description
DIG1_RX_LATE	23:16	0x4B	Rx Digital delay on PCIe Gen 1 in nsec units (75nsec). Software should program this register to 0x44.
DIG2_RX_LATE	31:24	0x27	Rx Digital delay on PCIe Gen 2 in nsec units (39nsec). Software should program this register to 0x1.

8.6.10 PCIe PHY Delay - PCIe_PHY_DELAY (0x12554; RW)

Reset by LAN_POWER_GOOD, PCIe Reset and software reset

Field	Bit(s)	Initial value	Description
PHY1_TX_LATE	7:0	0x2A	PCIe PHY Gen1 transmit typical delay in ns. Default = 42 nsec. Software should program this register to 0x0.
PHY2_TX_LATE	15:8	0x15	PCIe PHY Gen2 transmit typical delay in ns. Default = 21 nsec. Software should program this register to 0x0.
PHY1_RX_LATE	23:16	0x80	Average PCIe PHY Gen1 Receive delay in ns Default = 128 nsec. Software should program this register to 0x90.
PHY2_RX_LATE	31:24	0x40	Average PCIe PHY Gen2 Receive delay in ns Default = 64 nsec.

8.6.11 T1 on Timer 0 Low - PTM_T1_TIM0_L (0x12558; RO)

PTM_T1_TIM1_L, PTM_T1_TIM2_L and PTM_T1_TIM3_L are the same structure as PTM_T1_TIM0_L for timers 1, 2 and 3 respectively.

Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
T1_on_Timer0_L	~~31:0~~ 29:0	0x0	Low portion of the sampled value of the 1588 timer 0 at PTM T1 phase.
Reserved	31:30	0x0	Reserved

8.6.12 T1 on Timer 0 High - PTM_T1_TIM0_H (0x1255C; RO)

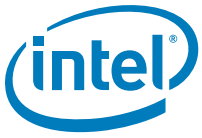
PTM_T1_TIM1_H, PTM_T1_TIM2_H and PTM_T1_TIM3_H are the same structure as PTM_T1_TIM0_H for timers 1, 2 and 3 respectively.

Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
T1_on_Timer0_H	31:0	0x0	High portion of the sampled value of the 1588 timer 0 at PTM T1 phase.

8.6.13 T4 minus T1 on Previous PTM Cycle - PTM_Prev_T4m1 (0x12578; RO)

Reset by LAN_POWER_GOOD and PCIe Reset



Field	Bit(s)	Initial value	Description
Prev_T4m1	31:0	0x0	Measured T4 minus T1 on the previous is PTM cycle.

8.6.14 T4 minus T1 on this PTM Cycle - PTM_Curr_T4m1 (0x1257C; RO)

Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
Curr_T4m1	31:0	0x0	Measured T4 minus T1 on this PTM cycle.

8.6.15 T3 minus T2 on Previous PTM Cycle - PTM_Prev_T3m2 (0x12580; RO)

Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
T3m2	31:0	0x0	Measured T3 minus T2 on the previous PTM cycle. The value in this register is extracted from the PTM ResponseD message received on the PCIe bus. This message is represented in little or big Endian notation depending on the system implementation. Foxville concludes the endian notation according to the TBD flags in the NVM.

8.6.16 T2 on Previous PTM Cycle Low - PTM_Prev_T2_L (0x12584; RO)

Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
Prev_T2_L	31:0	0x0	Low 32 bits of the sampled T2 time on the previous PTM cycle.

8.6.17 T2 on Previous PTM Cycle High - PTM_Prev_T2_H (0x12588; RO)

Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
Prev_T2_H	31:0	0x0	High 32 bits of the sampled T2 time on the previous PTM cycle.

8.6.18 T2 on this PTM Cycle Low - PTM_Curr_T2_L (0x1258C; RO)

Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
PTM_CURR_T2_L	31:0	0x0	Low 32 bits of the sampled T2 time on this PTM cycle.

8.6.19 T2 on this PTM Cycle High - PTM_Curr_T2_H (0x12590; RO)

Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
PTM_CURR_T2_H	31:0	0x0	High 32 bits of the sampled T2 time on this PTM cycle.

8.6.20 PTM PCIe Link Delay - PTM_TDELAY (0x12594; RO)

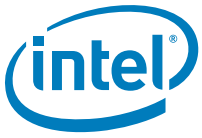
Reset by LAN_POWER_GOOD and PCIe Reset

Field	Bit(s)	Initial value	Description
T_DELAY	31:0	0x0	Calculated end to end PCIe delay. This register is viable only if the reported PTM_Prev_T3m2 is in little endian notation.

8.6.21 Time Params for L12 - PCIE_L12_TIMES (0x125A4; RW)

Timing parameters for L12.

Field	Bit(s)	Initial Value	Description
LTSSM_L12_T_PO	12:0	0x38	Wait time in L12.Entry before going to L12.Idle. The time it takes to the PCIE PHY to power off and go to P1.2 state, in clock cyles. Default 0x38 = 1.6uS. Loaded from the NVM word 0x52. This field must be set to the same value as the PHY_P11_2_P10.
RSRVD	15:13	0x0	
LTSSM_L12_STAY	28:16	0x78	time to stay in L12.Idle. The minimum time to stay at L1.2 once in it. In clock cycles. Default 0x78 = 4.15uS. Loaded from the NVM word 0x53.
RESRVED	31:29	0x0	



8.6.22 PCIE to P10 state - PCIE_TIMES_2P10 (0x125A8; RW)

The time required to move to p10 state in pcie phy.

Field	Bit(s)	Initial Value	Description
PHY_P11_2_P10	12:0	0x32	The warmup delay (as part of the L1.1 to L0 transition) is controlled by this field. This field is defined in 40nsec units with a HW default value equals to 2usec. This delay must be set to the same value as the LTSSM_L1EXTCLK_WARMUP_SCALE in the PCIE_L1_EXTCLK register and the L1 PM Sub states Control 2 Register in the PCIe configuration space. This field is loaded from the P11_2_P10 field in the NVM (set by default to 0xFC ~ 10usec)
RESERVED	15:13	0x0	
PHY_P12_2_P10	28:16	0x32	The warmup delay (as part of the L1.2 to L0 transition) is controlled by this field. This field is defined in 40nsec units with a HW default value equals to 2usec. This delay must be set to the same value as the PHY_P11_2_P10 in this register. This field is loaded from the P12_2_P10 field in the NVM (set by default to 0xFC ~ 10usec)
RESERVED	31:29	0x0	

8.6.23 PCIE Ext CLK Warm up - PCIE_L1_EXTCLK (0x125B0; RW)

Field	Bit(s)	Initial Value	Description
LTSSM_L1EXTCLK_WARMUP_SCALE	1:0	0x0	T_WARMUP Scale – Specifies the scale used for External clock warmup value. 00b = 2µs, 01b = 10µs, 10b = 100µs, 11b = Reserved. This field must be set to the same value as PHY_P11_2_P10 in the PCIE_TIMES_2P10 register as well as the L1 PM Sub states Control 2 Register in the PCIe configuration space.
Reserved	2	0x0	
LTSSM_L1EXTCLK_WARMUP_VALUE	7:3	0x1	T_WARMUP_VALUE sets the minimum amount of time (in µs multiply by scale) that the Port must wait in L1EXTCLK_WARMUP state.
RESERVED	31:8	0x0	

8.7 Semaphore Registers

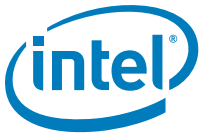
This section contains registers used to coordinate between firmware and software. The usage of these registers is described in Section 4.7.11.

8.7.1 Software Semaphore - SWSM (0x5B50; RW)

Field	Bit(s)	Initial Value	Description
SMBI (RS)	0	0b	Software/Software Semaphore Bit This bit is set by hardware when this register is read by the software device driver and cleared when the host driver writes a 0b to it. The first time this register is read, the value is 0b. In the next read the value is 1b (hardware mechanism). The value remains 1b until the software device driver clears it. This bit can be used as a semaphore between all Foxville driver threads. This bit is cleared on PCIe reset.
SWESMBI	1	0x0	Software/Firmware Semaphore Bit. This bit should be set only by the software device driver (read only to firmware). The bit is not set if bit zero in the FWSM register is set. The software device driver should set this bit and then read it to verify that it was set. If it was set, it means that the software device driver can access the SW_FW_SYNC register. The software device driver should clear this bit after modifying the SW_FW_SYNC register. ***Note:*** • If software takes ownership of the *SWSM.SWESMBI* bit for a duration longer than 10 ms, Firmware can take ownership of the bit. • Hardware clears this bit on a PCIe reset.
Reserved	30:2	0x0	Reserved. Write 0x0, ignore on read.

8.7.2 Firmware Semaphore - FWSM (0x5B54; RO to Host, RW to FW)

Field[1]	Bit(s)	Initial Value	Description
EEP_FW_Semaphore	0	0b	Software/Firmware Semaphore. Firmware should set this bit to 1b before accessing the *SW_FW_SYNC* register. If software is using the SWSM register and does not lock SW_FW_SYNC, firmware is able to set this bit to 1b. Firmware should set this bit back to 0b after modifying the *SW_FW_SYNC* register. ***Note:*** If software takes ownership of the *SWSM.SWESMBI* bit for a duration longer than 10 ms, firmware can take ownership of the bit.
FW_Mode	3:1	0x0	Firmware Mode. Indicates the firmware mode as follows: 000b = No manageability. Default mode for all SKUs. 001b = Foxville mode. A proxy code was loaded. 010b = PT mode. Foxville SKUs only. 011b = Reserved. 100b = Host interface only. In Foxville, this bit determines that a valid firmware code is running from the NVM but PT mode is disabled.
Reserved	5:4	00b	Reserved. Write 0x0, ignore on read.
EEP_Reload_Ind	6	0b	NVM Reloaded Indication. Set to 1b after firmware reloads the Firmware related sections of the NVM. Cleared by firmware at Firmware reset only.
Reserved	14:7	0x0	Reserved Write 0x0, ignore on read.



Field[1]	Bit(s)	Initial Value	Description
FW_Val_Bit	15	0b	Firmware Valid Bit. Hardware clears this bit in reset de-assertion so software can know firmware mode (bits 1-3) bits are invalid. In Foxville, firmware should set this bit to 1b when it is ready (end of boot sequence). Each time this bit is set to 1b, an ICR.MNG interrupt must be issued to host.
Reset_Cnt	18:16	0b	Reset Counter. Firmware increments the count on every firmware reset. After seven firmware reset events, the counter remains at seven and does not wrap around.
Ext_Err_Ind	24:19	0x0	External Error Indication Firmware writes here the reason that the firmware operation has stopped. For example, NVM CRC error, etc. Possible values: 0x00: No Error. 0x01: NVM CRC error in test configuration module. 0x02: NVM CRC Error in Loader Patch Module. 0x03: NVM CRC error in common firmware parameters module. 0x04: NVM CRC error in pass through. 0x05: Shadow RAM dump fault. 0x06: Bad Flash contents. 0x07: Reserved. 0x08: NVM CRC error in sideband configuration module. 0x09: NVM CRC error in flexible TCO filter configuration module. 0x0A: NVM CRC Error in NC-SI microcode download module. 0x0B: NVM CRC Error in NC-SI configuration module. 0x0C: NVM CRC Error in traffic type parameters module. 0x0D: NVM CRC Error in inventory NVM structure module. 0x0E: NVM CRC Error in PHY configuration structure module. 0x0F to 0x15: Reserved. 0x16: TLB table exceeded. 0x17: DMA load failed. 0x18: Reserved. 0x19: Flash device not supported. 0x1A: Invalid Flash checksum. 0x1B: Unspecified error. 0x1C to 0x1F: Reserved. 0x20: NVM CRC Error in hardware auto-load. 0x21: No manageability (No NVM). 0x22: TCO isolate mode active. 0x23: Management memory parity error. 0x24: Firmware NVM access failure. 0x25: Other management error detected. Cause can be found in ICR10 in Management AUX register. 0x26: IPG error detected at 2.5G link speed, downshift to 1G required 0x27 to 0x03F: Reserved ***Note:*** Following an error detection and *FWSM.Ext_Err_ind* update, the *ICR.MGMT* bit is set and an interrupt is sent to the host. However when values of 0x00 or 0x21 are placed in the *FWSM.Ext_Err_ind* field, the *ICR.MGMT* bit is not set and an interrupt is not generated.
PCIe_Config_Err_Ind	25	0b	PCIe Configuration Error Indication. Set to 1b by firmware when it fails to configure the PCIe interface. Cleared by firmware after successfully configuring the PCIe interface.

Field[1]	Bit(s)	Initial Value	Description
Reserved	26	0b	Reserved
Reserved	30:27	0b	Reserved. Write 0b, ignore on read.
Factory MAC address restored	31	0b	This bit is set if internal firmware restored the factory MAC address at power up or if the factory MAC address and the regular MAC address were the same.

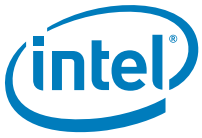
Notes:
1. This register should be written only by the manageability firmware. The software device driver should only read this register.
2. Firmware ignores the NVM semaphore in operating system hung states.
3. Bits 15:0 are cleared on firmware reset.

8.7.3 Software–Firmware Synchronization - SW_FW_SYNC (0x5B5C; RWM)

This register is intended to synchronize between software and firmware.

Note: If software takes ownership of bits in the *SW_FW_SYNC* register for a duration longer than 1 second, firmware can take ownership of the bit.

Field	Bit(s)	Initial Value	Description
SW_FLASH_SM	0	0b	When set to 1b, NVM access is owned by software.
SW_PHY_SM	1	0b	When set to 1b, PHY access is owned by software.
SW_I2C_SM	2	0b	When set to 1b, I^2C access register set (I2CCMD) is owned by software.
SW_MAC_CSR_SM	3	0b	When set to 1b, software owns access to shared CSRs. Note: Not in use.
Reserved	6:4	0x0	Reserved. Write 0x0, ignore on read.
SW_SVR_SM	7	0b	When set to 1b, the SVR/LVR control registers are owned by the software device driver.
SW_MB_SM	8	0b	When Set to 1b, the *SWMBWR* mailbox write register, is owned by the software device driver.
Reserved	9	0b	Reserved. Write 0b, ignore on read.
SW_MNG_SM	10	0b	When set to 1b, the management host interface is owned by the port driver. This bit can be used by the port driver when updating teaming or proxying information.
Reserved	15:11	0x0	Reserved. Write 0x0, ignore on read.
FW_FLASH_SM	16	0b	When set to 1b, NVM access is owned by firmware.
FW_PHY_SM	17	0b	When set to 1b, PHY access is owned by firmware.
FW_I2C_SM	18	0b	When set to 1b, I^2C access register set (I2CCMD) is owned by firmware.
FW_MAC_CSR_SM	19	0b	When set to 1b, firmware owns access to shared CSRs. Note: Not in use.



Field	Bit(s)	Initial Value	Description
Reserved	22:20	0b	Reserved. Write 0x0, ignore on read.
FW_SVR_SM	23	0b	When set to 1b, the SVR/LVR control registers are owned by firmware.
Reserved	31:24	0x0	Reserved Write 0x0, ignore on read.

Reset conditions:

— The software-controlled bits 15:0 are reset as any other CSR on software resets, D3hot exit and Forced TCO. Software is expected to clear the bits on entry to D3 state.

— The firmware controlled bits (bits 31:16) are reset on LAN_PWR_GOOD (power up) and firmware reset.

8.8 Interrupt Register Descriptions

8.8.1 Extended Interrupt Cause - EICR (0x1580; RC/W1C)

This register contains the frequent interrupt conditions for Foxville. Each time an interrupt causing event occurs, the corresponding interrupt bit is set in this register. An interrupt is generated each time one of the bits in this register is set and the corresponding interrupt is enabled via the Interrupt Mask Set/Read register. The interrupt might be delayed by the selected Interrupt Throttling register.

Note that the software device driver cannot determine from the RxTxQ bits what was the cause of the interrupt. The possible causes for asserting these bits are: Receive descriptor write back, receive descriptor minimum threshold hit, low latency interrupt for Rx, and transmit descriptor write back.

Writing a 1b to any bit in the register clears that bit. Writing a 0b to any bit has no effect on that bit.

Register bits are cleared on register read if GPIE.Multiple_MSIX = 0b.

Auto clear can be enabled for any or all of the bits in this register.

Table 8-9. EICR Register - Non-MSI-X Mode (GPIE.Multiple_MSIX = 0b)

Field	Bit(s)	Initial Value	Description
RxTxQ	3:0	0x0	Receive/Transmit Queue Interrupts. One bit per queue or a bundle of queues, activated on receive/transmit queue events for the corresponding bit, such as: • Receive descriptor write back • Receive descriptor minimum threshold hit • Transmit descriptor write back. The mapping of the actual queue to the appropriate *RxTxQ* bit is according to the IVAR registers.
Reserved	29:4	0x0	Reserved. Write 0x0, ignore on read.
TCP Timer	30	0b	TCP Timer Expired. Activated when the TCP timer reaches its terminal count.
Other Cause	31	0b	Interrupt Cause Active. Activated when any bit in the ICR register is set.

Note: Bits are not reset by device reset (*CTRL.DEV_RST*).

Table 8-10. EICR Register - MSI-X Mode (GPIE.Multiple_MSIX = 1b)

Field	Bit(s)	Initial Value	Description
MSIX	4:0	0x0	Indicates an interrupt cause mapped to MSI-X vectors 4:0. ***Note:*** Bits are not reset by device reset (*CTRL.DEV_RST*).
Reserved	31:5	0x0	Reserved. Write 0x0, ignore on read.

8.8.2 Extended Interrupt Cause Set - EICS (0x1520; WO)

Software uses this register to set an interrupt condition. Any bit written with a 1b sets the corresponding bit in the Extended Interrupt Cause Read register. An interrupt is then generated if one of the bits in this register is set and the corresponding interrupt is enabled via the Extended Interrupt Mask Set/Read register. Bits written with 0b are unchanged.

Table 8-11. EICS Register - Non MSI-X mode (GPIE.Multiple_MSIX = 0b)

Field	Bit(s)	Initial Value	Description
RxTxQ	3:0	0x0	Sets to corresponding EICR RxTXQ interrupt condition.
Reserved	29:4	0x0	Reserved. Write 0x0, ignore on read.
TCP Timer	30	0b	Sets the corresponding EICR TCP timer interrupt condition.
Reserved	31	0b	Reserved. Write 0b, ignore on read.

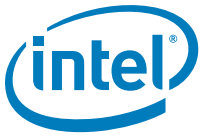
Note: In order to set bit 31 of the EICR (Other Causes), the ICS and IMS registers should be used in order to enable one of the legacy causes.

Table 8-12. EICS Register - MSI-X Mode (GPIE.Multiple_MSIX = 1b)

Field	Bit(s)	Initial Value	Description
MSI-X	4:0	0x0	Sets the corresponding *EICR* bit of MSI-X vectors 4:0
Reserved	31:5	0x0	Reserved. Write 0x0, ignore on read.

8.8.3 Extended Interrupt Mask Set/Read - EIMS (0x1524; RWM)

Reading this register returns which bits that have an interrupt mask set. An interrupt in *EICR* is enabled if its corresponding mask bit is set to 1b and disabled if its corresponding mask bit is set to 0b. A PCI interrupt is generated each time one of the bits in this register is set and the corresponding interrupt condition occurs (subject to throttling). The occurrence of an interrupt condition is reflected by having a bit set in the Extended Interrupt Cause Read register.



An interrupt might be enabled by writing a 1b to the corresponding mask bit location (as defined in the *EICR* register) in this register. Any bits written with a 0b are unchanged. As a result, if software needs to disable an interrupt condition that had been previously enabled, it must write to the *Extended Interrupt Mask Clear* register rather than writing a 0b to a bit in this register.

Table 8-13. EIMS Register - Non-MSI-X Mode (GPIE.Multiple_MSIX = 0b)

Field	Bit(s)	Initial Value	Description
RxTxQ	3:0	0x0	Set the *Mask* bit for the corresponding EICR RxTXQ interrupt.
Reserved	29:4	0x0	Reserved. Write 0x0, ignore on read.
TCP Timer	30	0b	Set the *Mask* bit for the corresponding EICR TCP timer interrupt condition.
Other Cause	31	1b	Set the *Mask* bit for the corresponding EICR other cause interrupt condition.

Note: Bits are not reset by device reset (*CTRL.DEV_RST*).

Table 8-14. EIMS Register - MSI-X Mode (GPIE.Multiple_MSIX = 1b)

Field	Bit(s)	Initial Value	Description
MSI-X	4:0	0x0	Set the *Mask* bit for the corresponding *EICR* bit of the MSI-X vectors 4:0. ***Note:*** Bits are not reset by device reset (*CTRL.DEV_RST*).
Reserved	31:5	0x0	Reserved Write 0x0, ignore on read.

8.8.4 Extended Interrupt Mask Clear - EIMC (0x1528; WO)

This register provides software a way to disable certain or all interrupts. Software disables a given interrupt by writing a 1b to the corresponding bit in this register.

On interrupt handling, the software device driver should set all the bits in this register related to the current interrupt request even though the interrupt was triggered by part of the causes that were allocated to this vector.

Interrupts are presented to the bus interface only when the mask bit is set to 1b and the cause bit is set to 1b. The status of the mask bit is reflected in the Extended Interrupt Mask Set/Read register and the status of the cause bit is reflected in the Interrupt Cause Read register.

Software blocks interrupts by clearing the corresponding mask bit. This is accomplished by writing a 1b to the corresponding bit location (as defined in the EICR register) of that interrupt in this register. Bits written with 0b are unchanged (their mask status does not change).

Table 8-15. EIMC Register - Non-MSI-X Mode (GPIE.Multiple_MSIX = 0b)

Field	Bit(s)	Initial Value	Description
RxTxQ	3:0	0x0	Clear the *Mask* bit for the corresponding *EICR* RxTXQ interrupt.
Reserved	29:4	0x0	Reserved. Write 0x0, ignore on read.
TCP Timer	30	0b	Clear the *Mask* bit for the corresponding *EICR* TCP timer interrupt.
Other Cause	31	1b	Clear the *Mask* bit for the corresponding *EICR* other cause interrupt.

Table 8-16. EIMC Register - MSI-X Mode (GPIE.Multiple_MSIX = 1b)

Field	Bit(s)	Initial Value	Description
MSI-X	4:0	0x0	Clear the *Mask* bit for the corresponding *EICR* bit of MSI-X vectors 4:0.
Reserved	31:5	0x0	Reserved. Write 0x0, ignore on read.

8.8.5 Extended Interrupt Auto Clear - EIAC (0x152C; RW)

This register is mapped like the EICS, EIMS, and EIMC registers, with each bit mapped to the corresponding MSI-X vector.

This register is relevant to MSI-X mode only, where read-to-clear can not be used, as it might erase causes tied to other vectors. If any bits are set in EIAC, the EICR register should not be read. Bits without auto clear set, need to be cleared with write-to-clear.

Note: *EICR* bits that have auto clear set are cleared by the internal emission of the corresponding MSI-X message even if this vector is disabled by the operating system.

The MSI-X message can be delayed by *EITR* moderation from the time the *EICR* bit is activated.

Table 8-17. EIAC Register

Field	Bit(s)	Initial Value	Description
MSI-X	4:0	0x0	Auto clear bit for the corresponding *EICR* bit of the MSI-X vectors 4:0. ***Notes:*** • Bits are not reset by device reset (*CTRL.DEV_RST*). • When *GPIE.Multiple_MSIX* = 0b (Non-MSI-X Mode) bits 8 and 9 are read only and should be ignored.
Reserved	31:5	0x0	Reserved. Write 0x0, ignore on read.

8.8.6 Extended Interrupt Auto Mask Enable - EIAM (0x1530; RW)

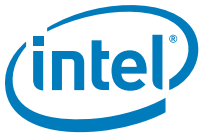
Each bit in this register enables clearing of the corresponding bit in EIMS register following read- or write-to-clear to EICR or setting of the corresponding bit in EIMS following a write-to-set to EICS.

In MSI-X mode, this register controls which of the bits in the EIMS register to clear upon interrupt generation if enabled via the *GPIE.EIAME* bit.

Note: When operating in MSI mode and setting any bit in the EIAM register causes the clearing of all bits in the EIMS register and the masking of all interrupts after generating a MSI interrupt.

Table 8-18. EIAM Register - Non-MSI-X Mode (GPIE.Multiple_MSIX = 0b)

Field	Bit(s)	Initial Value	Description
RxTxQ	3:0	0x0	*Auto Mask* bit for the corresponding *EICR RxTxQ* interrupt.
Reserved	29:4	0x0	Reserved. Write 0x0, ignore on read.
TCP Timer	30	0b	*Auto Mask* bit for the corresponding *EICR TCP timer* interrupt condition.
Other Cause	31	0b	*Auto Mask* bit for the corresponding *EICR other cause* interrupt condition.



Note: Bits are not reset by device reset (*CTRL.DEV_RST*).

Table 8-19. EIAM Register - MSI-X Mode (GPIE.Multiple_MSIX = 1b)

Field	Bit(s)	Initial Value	Description
MSIX	4:0	0x0	*Auto Mask* bit for the corresponding *EICR* bit of MSI-X vectors 4:0. ***Note:*** Bits are not reset by device reset (*CTRL.DEV_RST*).
Reserved	31:5	0x0	Reserved. Write 0x0, ignore on read.

8.8.7 Interrupt Cause Read Register - ICR (0x1500; RC/W1C)

This register contains the interrupt conditions for Foxville that are not present directly in the EICR. Each time an ICR interrupt causing event occurs, the corresponding interrupt bit is set in this register. The *EICR.Other* bit reflects the setting of interrupt causes from *ICR* as masked by the Interrupt Mask Set/ Read register. Each time all un-masked causes in *ICR* are cleared, the *EICR.Other* bit is also cleared.

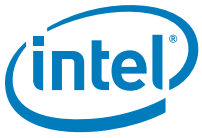
ICR bits are cleared on register read. Clear-on-read can be enabled/disabled through a general configuration register bit. Refer to Section 7.3.3 for additional information.

Auto clear is not available for the bits in this register.

In order to prevent unwanted Link Status Change (LSC) interrupts during initialization, software should disable this interrupt until the end of initialization.

Field	Bit(s)	Initial Value	Description
TXDW	0	0b	Transmit Descriptor Written Back. Set when Foxville writes back a Tx descriptor to memory.
Reserved	1	0b	Reserved. Write 0x0, ignore on read.
LSC	2	0b	Link Status Change. This bit is set each time the link status changes (either from up to down, or from down to up). This bit is affected by the LINK indication from the PHY (internal PHY mode).
Reserved	3	0b	Reserved. Write 0x0, ignore on read.
RXDMT0	4	0b	Receive Descriptor Minimum Threshold Reached. Indicates that the minimum number of receive descriptors are available and software should load more receive descriptors.
Reserved	5	0b	Reserved. Write 0x0, ignore on read.
Rx Miss	6	0b	Missed packet interrupt is activated for each received packet that overflows the Rx packet buffer (overrun). Note that the packet is dropped and also increments the associated MPC counter. ***Note:*** Could be caused by no available receive buffers or because PCIe receive bandwidth is inadequate.
RXDW	7	0b	Receiver Descriptor Write Back. Set when Foxville writes back an Rx descriptor to memory.
Reserved	8	0b	Reserved. Write 0x0, ignore on read.
MDAC	9	0b	MDIO Access Complete Set when an MDIO access completes.
GPHY	10	0b	Internal PHY interrupt.

Field	Bit(s)	Initial Value	Description
GPI_SDP0	11	0b	General Purpose Interrupt on SDP0. If GPI interrupt detection is enabled on this pin (via *CTRL.SDP0_GPIEN*), this interrupt cause is set when the SDP0 is sampled high.
GPI_SDP1	12	0b	General Purpose Interrupt on SDP1. If GPI interrupt detection is enabled on this pin (via *CTRL.SDP1_GPIEN*), this interrupt cause is set when the SDP1 is sampled high.
GPI_SDP2	13	0b	General Purpose Interrupt on SDP2. If GPI interrupt detection is enabled on this pin (via *CTRL_EXT.SDP2_GPIEN*), this interrupt cause is set when the SDP2 is sampled high.
GPI_SDP3	14	0b	General Purpose Interrupt on SDP3. If GPI interrupt detection is enabled on this pin (via *CTRL_EXT.SDP3_GPIEN*), this interrupt cause is set when the SDP3 is sampled high.
PTRAP	15	0b	Probe trap interrupt - when set, the probe mode trap test mode trapped the requested event.
Reserved	17:16	00b	Reserved. Write 0x0, ignore on read.
MNG	18	0b	Manageability Event Detected. Indicates that a manageability event happened. When bit is set due to detection of error by management, *FWSM.Ext_Err_Ind* field is updated with the error cause.
Time_Sync	19	0b	Time_Sync Interrupt. This interrupt cause is set if the interrupt is generated by the Time Sync interrupt (See *TSICR* and *TSIM* registers).
Reserved	21:20	0b	Reserved. Write 0x0, ignore on read.
FER	22	0b	Fatal Error. This bit is set when a fatal error is detected in one of the memories.
Reserved	23	0b	Reserved. Write 0x0, ignore on read.
PCI Exception	24	0b	The PCI timeout exception is activated by one of the following events when the specific PCI event is reported in the PICAUSE register and the appropriate bit in the PIENA register is set: 1. I/O completion abort. 2. Unsupported I/O request (wrong address). 3. Byte-enable error - Access to the client that does not support partial BE access (All but Flash, MSIX and the PCIe target). 4. Timeout occurred in the FUNC block. 5. BME of the PF is cleared.
SCE	25	0b	DMA Coalescing Clock Control Event. This bit is set when the multicast or broadcast DMA coalescing clock control mechanism is activated or de-activated.
Software WD	26	0b	Software Watchdog. This bit is set after a software watchdog timer times out.
Reserved	27	0b	Reserved. Write 0x0, ignore on read.
MDDET	28	0b	Detected Malicious driver behavior Occurs when one of the queues used malformed descriptors. **Note:** This bit should never rise during normal operation.



Field	Bit(s)	Initial Value	Description
TCP Timer	29	0b	TCP Timer Interrupt. Activated when the TCP timer reaches its terminal count.
DRSTA	30	0b	Device Reset Asserted. Indicates *CTRL.DEV_RST* was asserted. When a device reset occurs, the port should re-initialize registers and descriptor rings. ***Note:*** This bit is not reset by device reset (*CTRL.DEV_RST*).
INTA	31	0b	Interrupt Asserted. Indicates that the INT line is asserted. Can be used by the software device driver in a shared interrupt scenario to decide if the received interrupt was emitted by Foxville. This bit is not valid in MSI/MSI-X environments.

8.8.8 Interrupt Cause Set Register - ICS (0x1504; WO)

Software uses this register to set an interrupt condition. Any bit written with a 1b sets the corresponding interrupt. This results in the corresponding bit being set in the Interrupt Cause Read Register (refer to Section 8.8.7). A PCIe interrupt is generated if one of the bits in this register is set and the corresponding interrupt is enabled through the Interrupt Mask Set/Read Register (refer to Section 8.8.9). Bits written with 0b are unchanged. Refer to Section 7.3.3 for additional information.

Field	Bit(s)	Initial Value	Description
TXDW	0	0b	Sets the Transmit Descriptor Written Back Interrupt.
Reserved	1	0b	Reserved. Write 0b, ignore on read.
LSC	2	0b	Sets the Link Status Change Interrupt.
Reserved	3	0b	Reserved. Write 0x0, ignore on read.
RXDMT0	4	0b	Sets the Receive Descriptor Minimum Threshold Hit Interrupt.
Reserved	5	0b	Reserved. Write 0x0, ignore on read.
Rx Miss	6	0b	Sets the Rx Miss Interrupt.
RXDW	7	0b	Sets the Receiver Descriptor Write Back Interrupt.
Reserved	8	0b	Reserved. Write 0b, ignore on read.
MDAC	9	0b	Sets the MDI/O Access Complete Interrupt.
GPHY	10	0b	Sets the internal PHY interrupt.
GPI_SDP0	11	0b	Sets the General Purpose interrupt, related to SDP0 pin.
GPI_SDP1	12	0b	Sets the General Purpose interrupt, related to SDP1 pin.
GPI_SDP2	13	0b	Sets the General Purpose interrupt, related to SDP2 pin.
GPI_SDP3	14	0b	Sets the General Purpose interrupt, related to SDP3 pin.
PTRAP	15	0b	Set the Probe trap interrupt
Reserved	17:16	0x0	Reserved. Write 0x0, ignore on read.
MNG	18	0b	Sets the Management Event Interrupt.
Time_Sync	19	0b	Sets the Time_Sync interrupt.
Reserved	21:20	0x0	Reserved. Write 0x0, ignore on read.
FER	22	0b	Sets the Fatal Error interrupt.
Reserved	23	0b	Reserved. Write 0b, ignore on read.

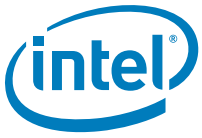
Field	Bit(s)	Initial Value	Description
PCI Exception	24	0b	Sets the PCI Exception interrupt.
SCE	25	0b	Sets the DMA Coalescing Clock Control Event interrupt.
Software WD	26	0b	Sets the Software Watchdog interrupt.
Reserved	27	0b	Reserved. Write 0b, ignore on read.
MDDET	28	0b	Sets the Detected Malicious driver behavior Interrupt.
TCP Timer	29	0b	Sets the TCP timer interrupt.
DRSTA	30	0b	Sets the Device Reset Asserted Interrupt. Note that when setting this bit a DRSTA interrupt is generated on this port only.
Reserved	31	0b	Reserved. Write 0b, ignore on read.

8.8.9 Interrupt Mask Set/Read Register - IMS (0x1508; RW)

Reading this register returns bits that have an interrupt mask set. An interrupt is enabled if its corresponding mask bit is set to 1b and disabled if its corresponding mask bit is set to 0b. A PCIe interrupt is generated each time one of the bits in this register is set and the corresponding interrupt condition occurs. The occurrence of an interrupt condition is reflected by having a bit set in the Interrupt Cause Read register (refer to Section 8.8.7).

A particular interrupt can be enabled by writing a 1b to the corresponding mask bit in this register. Any bits written with a 0b are unchanged. As a result, if software desires to disable a particular interrupt condition that had been previously enabled, it must write to the Interrupt Mask Clear Register (refer to Section 8.8.10) rather than writing a 0b to a bit in this register. Refer to Section 7.3.3 for additional information.

Field	Bit(s)	Initial Value	Description
TXDW	0	0b	Sets/reads the mask for Transmit Descriptor Written Back interrupt.
Reserved	1	0b	Reserved. Write 0b, ignore on read.
LSC	2	0b	Sets/Reads the mask for Link Status Change interrupt.
Reserved	3	0b	Reserved. Write 0b, ignore on read.
RXDMT0	4	0b	Sets/reads the mask for Receive Descriptor Minimum Threshold Hit interrupt.
Reserved	5	0b	Reserved. Write 0b, ignore on read.
Rx Miss	6	0b	Sets/reads the mask for the Rx Miss interrupt.
RXDW	7	0b	Sets/reads the mask for Receiver Descriptor Write Back interrupt.
Reserved	8	0b	Reserved. Write 0b, ignore on read.
MDAC	9	0b	Sets/Reads the mask for MDIO Access Complete Interrupt.
GPHY	10	0b	Sets/Reads the mask for Internal PHY interrupt.
GPI_SDP0	11	0b	Sets/Reads the mask for General Purpose Interrupt, related to SDP0 pin.
GPI_SDP1	12	0b	Sets/Reads the mask for General Purpose Interrupt, related to SDP1 pin.
GPI_SDP2	13	0b	Sets/Reads the mask for General Purpose Interrupt, related to SDP2 pin.
GPI_SDP3	14	0b	Sets/Reads the mask for General Purpose Interrupt, related to SDP3 pin.
PTRAP	15	0b	Sets/reads the mask for the Probe trap interrupt



Field	Bit(s)	Initial Value	Description
Reserved	17:16	0x0	Reserved. Write 0x0, ignore on read.
MNG	18	0b	Sets/reads the mask for Management Event interrupt.
Time_Sync	19	0b	Sets/reads the mask for Time_Sync interrupt.
Reserved	20	0b	Reserved. Write 0b, ignore on read.
Reserved	21	0b	Reserved. Write 0b, ignore on read.
FER	22	0b	Sets/reads the mask for the Fatal Error interrupt.
Reserved	23	0b	Reserved. Write 0b, ignore on read.
PCI Exception	24	0b	Sets/reads the mask for the PCI Exception interrupt.
SCE	25	0b	Sets/reads the mask for the DMA Coalescing Clock Control Event interrupt.
Software WD	26	0b	Sets/reads the mask for the Software Watchdog interrupt.
Reserved	27	0b	Reserved. Write 0b, ignore on read.
MDDET	28	0b	Sets/Reads the mask for Detected Malicious driver behavior Interrupt.
TCP Timer	29	0b	Sets/reads the mask for TCP timer interrupt.
DRSTA	30	0b	Sets/reads the mask for Device Reset Asserted interrupt. ***Note:*** Bit is not reset by device reset (CTRL.DEV_RST).
Reserved	31	0b	Reserved. Write 0b, ignore on read.

8.8.10 Interrupt Mask Clear Register - IMC (0x150C; WO)

Software uses this register to disable an interrupt. Interrupts are presented to the bus interface only when the mask bit is set to 1b and the cause bit set to 1b. The status of the mask bit is reflected in the Interrupt Mask Set/Read register (refer to Section 8.8.9), and the status of the cause bit is reflected in the Interrupt Cause Read register (refer to Section 8.8.7). Reading this register returns the value of the IMS register.

Software blocks interrupts by clearing the corresponding mask bit. This is accomplished by writing a 1b to the corresponding bit in this register. Bits written with 0b are unchanged (their mask status does not change).

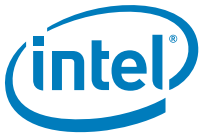
Software device driver should set all the bits in this register related to the current interrupt request when handling interrupts, even though the interrupt was triggered by part of the causes that were allocated to this vector. Refer to Section 7.3.3 for additional information.

Field	Bit(s)	Initial Value	Description
TXDW	0	0b	Clears the mask for Transmit Descriptor Written Back interrupt.
Reserved	1	0b	Reserved. Write 0b, ignore on read.
LSC	2	0b	Clears the mask for Link Status Change interrupt.
Reserved	3	0b	Reserved. Write 0b, ignore on read.
RXDMT0	4	0b	Clears the mask for Receive Descriptor Minimum Threshold Hit interrupt.

Field	Bit(s)	Initial Value	Description
Reserved	5	0b	Reserved. Write 0b, ignore on read.
Rx Miss	6	0b	Clears the mask for the Rx Miss interrupt.
RXDW	7	0b	Clears the mask for the Receiver Descriptor Write Back interrupt.
Reserved	8	0b	Reserved. Write 0b, ignore on read.
MDAC	9	0b	Clears the mask for the MDIO Access Complete Interrupt.
GPHY	10	0b	Clears the mask for the Internal PHY interrupt.
GPI_SDP0	11	0b	Clears the mask for the General Purpose interrupt, related to SDP0 pin.
GPI_SDP1	12	0b	Clears the mask for the General Purpose interrupt, related to SDP1 pin.
GPI_SDP2	13	0b	Clears the mask for the General Purpose interrupt, related to SDP2 pin.
GPI_SDP3	14	0b	Clears the mask for the General Purpose interrupt, related to SDP3 pin.
PTRAP	15	0	Clears the mask for the Probe trap interrupt
Reserved	17:16	0x0	Reserved. Write 0x0, ignore on read.
MNG	18	0b	Clears the mask for the Management Event interrupt.
Time_Sync	19	0b	Clears the mask for the Time_Sync interrupt.
Reserved	20	0b	Reserved. Write 0b, ignore on read.
Reserved	21	0b	Reserved. Write 0b, ignore on read.
FER	22	0b	Clears the mask for the Fatal Error interrupt.
Reserved	23	0b	Reserved. Write 0b, ignore on read.
PCI Exception	24	0b	Clears the mask for the PCI Exception interrupt.
SCE	25	0b	Clears the mask for the DMA Coalescing Clock Control Event interrupt.
Software WD	26	0b	Clears the mask for Software Watchdog Interrupt.
Reserved	27	0b	Reserved. Write 0b, ignore on read.
MDDET	28	0b	Clears the mask for Detected Malicious driver behavior Interrupt.
TCP timer	29	0b	Clears the mask for TCP timer interrupt.
DRSTA	30	0b	Clears the mask for Device Reset Asserted interrupt.
Reserved	31	0b	Reserved. Write 0b, ignore on read.

8.8.11 Interrupt Acknowledge Auto Mask Register - IAM (0x1510; RW)

Field	Bit(s)	Initial Value	Description
IAM_VALUE	30:0	0x0	An *ICR* read or write has the side effect of writing the contents of this register to the IMC register. If *GPIE.NSICR* = 0b, then the copy of this register to the IMC register occurs only if at least one bit is set in the IMS register and there is a true interrupt as reflected in the *ICR.INTA* bit. Refer to Section 7.3.3 for additional information. ***Note:*** Note: Bit 30 of this register is not reset by device reset (*CTRL.DEV_RST*).
Reserved	31	0b	Reserved. Write 0b, ignore on read.



8.8.12 Interrupt Throttle - EITR (0x1680 + 4*n [n = 0...4]; RW)

Each EITR is responsible for an interrupt cause (RxTxQ, TCP timer and Other Cause). The allocation of EITR-to-interrupt cause is through the IVAR registers.

Software uses this register to pace (or even out) the delivery of interrupts to the host processor. This register provides a guaranteed inter-interrupt delay between interrupts asserted by Foxville, regardless of network traffic conditions. To independently validate configuration settings, software can use the following algorithm to convert the inter-interrupt interval value to the common interrupts/sec. performance metric:

$$\text{interrupts/sec} = (1 * 10^{-6}\text{sec} \times \text{interval})^{-1}$$

A counter counts in units of $1*10^{-6}$ sec. After counting interval number of units, an interrupt is sent to the software. The previous equation gives the number of interrupts per second. The equation that follows is the time in seconds between consecutive interrupts.

For example, if the interval is programmed to 125 (decimal), Foxville guarantees the processor does not receive an interrupt for 125 µs from the last interrupt. The maximum observable interrupt rate from Foxville should never exceed 8000 interrupts/sec.

Inversely, inter-interrupt interval value can be calculated as:

$$\text{inter-interrupt interval} = (1 * 10^{-6}\text{sec} \times \text{interrupt/sec})^{-1}$$

The optimum performance setting for this register is very system and configuration specific. An initial suggested range is 2 to 175 (0x02 to 0xAF).

Note: Setting EITR to a non-zero value can cause an interrupt cause Rx/Tx statistics miscount.

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0x0	Reserved. Write 0x0, ignore on read.
Interval	14:2	0x0	Minimum Inter-interrupt Interval. The interval is specified in 1 µs increments. A null value is not a valid setting.
LLI_EN	15	0b	LLI moderation enable.
LL Counter (RWM)	20:16	0x0	Reflects the current credits for that EITR for LL interrupts. If the *CNT_INGR* is not set, this counter can be directly written by software at any time to alter the throttles performance
Moderation Counter (RWM)	30:21	0x0	Down counter, exposes only the 10 most significant bits of the real 12-bit counter. Loaded with interval value each time the associated interrupt is signaled. Counts down to zero and stops. The associated interrupt is signaled each time this counter is zero and an associated (via the Interrupt Select register) *EICR* bit is set. If the CNT_INGR is not set, this counter can be directly written by software at any time to alter the throttles performance.
CNT_INGR (WO)	31	0b	When set, hardware does not override the counters fields (ITR counter and LLI credit counter), so they keep their previous value. Relevant for the current write only and is always read as zero.

Note: The EITR register and interrupt mechanism is not reset by device reset (*CTRL.DEV_RST*). Occurrence of device reset interrupt causes immediate generation of all pending interrupts.

8.8.13 Interrupt Vector Allocation Registers - IVAR (0x1700 + 4*n [n=0...1]; RW)

These registers have two modes of operation:

1. In MSI-X mode, these registers define the allocation of the different interrupt causes as defined in Table 7-51 to one of the MSI-X vectors. Each *INT_Alloc[i]* (i=0...7) field is a byte indexing an entry in the MSI-X Table Structure and MSI-X PBA Structure.
2. In non MSI-X mode, these registers define the allocation of the Rx and Tx queues interrupt causes to one of the RxTxQ bits in the *EICR* register. Each *INT_Alloc[i]* (i=...7) field is a byte indexing the appropriate RxTxQ bit as defined in Table 7-50.

Field	Bit(s)	Initial Value	Description
INT_Alloc[4*n]	2:0	0x0	Defines the MSI-X vector assigned to Rx0 or Rx2 for IVAR[0] or IVAR[1], respectively. Valid values are 0 to 4 for MSI-X mode and 0 to 3 in non-MSI-X mode.
Reserved	6:3	0x0	Reserved. Write 0x0, ignore on read.
INT_Alloc[4*n]	7	0b	Valid bit for INT_Alloc[4*n].
INT_Alloc[4*n+1]	10:8	0x0	Defines the MSI-X vector assigned to Tx0 or Tx2 for IVAR[0] or IVAR[1], respectively. Valid values are 0 to 4 for MSI-X mode and 0 to 3 in non-MSI-X mode.
Reserved	14:11	0x0	Reserved. Write 0x0, ignore on read.
INT_Alloc[4*n+1]	15	0b	Valid bit for INT_Alloc[4*n+1].
INT_Alloc[4*n+2]	18:16	0x0	Defines the MSI-X vector assigned to Rx1 or Rx3 for IVAR[0] or IVAR[1], respectively. Valid values are 0 to 4 for MSI-X mode and 0 to 3 in non-MSI-X mode.
Reserved	22:19	0x0	Reserved. Write 0x0, ignore on read.
INT_Alloc[4*n+2]	23	0b	Valid bit for INT_Alloc[4*n+2].
INT_Alloc[4*n+3]	26:24	0x0	Defines the MSI-X vector assigned to Tx1 or Tx3 for IVAR[0] or IVAR[1], respectively. Valid values are 0 to 4 for MSI-X mode and 0 to 3 in non-MSI-X mode.
Reserved	30:27	0x0	Reserved. Write 0x0, ignore on read.
INT_Alloc[4*n+3]	31	0b	Valid bit for INT_Alloc[4*n+3].

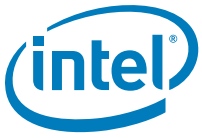
Note: If invalid values are written to the INT_Alloc fields the result is unexpected.

8.8.14 Interrupt Vector Allocation Registers - MISC IVAR_MISC (0x1740; RW)

This register is used only in MSI-X mode. This register defines the allocation of the Other Cause and TCP Timer interrupts to one of the MSI-X vectors.

Note: Bits mapped to Other Cause are not reset by Device Reset (*CTRL.DEV_RST*).

Field	Bit(s)	Initial Value	Description
INT_Alloc[8]	2:0	0x0	Defines the MSI-X vector assigned to the TCP timer interrupt cause. Valid values are 0 to 4.
Reserved	6:3	0x0	Reserved. Write 0x0, ignore on read.
INT_Alloc[8]	7	0b	Valid bit for INT_Alloc[8].



Field	Bit(s)	Initial Value	Description
INT_Alloc[9]	10:8	0x0	Defines the MSI-X vector assigned to the Other Cause interrupt. Valid values are 0 to 4.
Reserved	14:11	0x0	Reserved. Write 0x0, ignore on read.
INT_Alloc[9]	15	0b	Valid bit for INT_Alloc[9].
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.8.15 General Purpose Interrupt Enable - GPIE (0x1514; RW)

Register is reset by LAN_PWR_GOOD and PCIe Reset only.

Field	Bit(s)	Initial Value	Description
NSICR	0	0b	Non Selective Interrupt Clear on Read. When set, every read of ICR clears it. When this bit is cleared, an ICR read causes it to be cleared only if an actual interrupt was asserted or *IMS* = 0x0. Refer to Section 7.3.3 for additional information.
Reserved	3:1	0x0	Reserved. Write 0x0, ignore on read.
Multiple MSIX	4	0b	0b = In MSI or MSI-X mode, with a single vector, *IVAR* maps Rx/Tx causes to 4 EICR bits but MSIX[0] is asserted for all. 1b = MSIX mode, *IVAR* maps Rx/Tx causes, TCP Timer and Other Cause interrupts to 5 MSI-x vectors reflected in 5 *EICR* bits. ***Note:*** When set, the EICR register is not cleared on read.
Reserved	6:5	0x0	Reserved. Write 0x0, ignore on read.
LL Interval	11:7	0x0	Low Latency Credits Increment Rate. The interval is specified in 4 µs increments. ***Note:*** When LLI moderation is enabled (LLI_EN bit set), this filed shall be set with a value different than 0x0.
Reserved	29:12	0x0	Reserved. Write 0x0, ignore on read.
EIAME	30	0b	Extended Interrupt Auto Mask Enable. When set (usually in MSI-X mode) and after sending a MSI-X message, if bits in the EIAM register associated with this message are set, then the corresponding bits in the EIMS register are cleared. Otherwise, EIAM is used only after reading or writing the EICR/EICS registers. ***Note:*** When this bit is set in MSI mode, setting of any bit in the EIAM register causes the clearing of all bits in the EIMS register and masking of all interrupts after generating a MSI interrupt.
PBA_support	31	0b	PBA Support. When set, setting one of the extended interrupts masks via EIMS causes the *PBA* bit of the associated MSI-X vector to be cleared. Otherwise, Foxville behaves in a way that supports legacy INT-x interrupts. ***Note:*** Should be cleared when working in INT-x or MSI mode and set in MSI-X mode.

8.8.16 PCIe Interrupt Cause - PICAUSE (0x5B88; RW1/C)

Reset by LAN_PWR_GOOD, PCIe reset and on move from D3 to D0 device state.

Field	Bit(s)	Init.	Description
CA	0	0b	PCI Completion Abort Exception Issued.
UA	1	0b	Reserved. Write 0x0, ignore on read.
BE	2	0b	Wrong byte-enable exception in the FUNC unit.
TO	3	0b	PCI timeout exception in the FUNC unit.
BMEF	4	0b	Asserted when Bus-Master-Enable (BME) of the PF is de-asserted.
ABR	5	0b	PCI Completer Abort Received. PCI Completer Abort (CA) or Unsupported Request (UR) received (set after receiving CA or UR). ***Note:*** When this bit is set, all PCIe master activity is stopped. Software should issue a software (*CTRL.DEV_RST*) reset to enable PCIe activity.
Reserved	31:6	0x0	Reserved. Write 0x0, ignore on read.

8.8.17 PCIe Interrupt Enable - PIENA (0x5B8C; RW)

Reset by LAN_PWR_GOOD, PCIe Reset and on move from D3 to D0 device state.

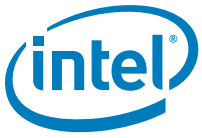
Field	Bit(s)	Init.	Description
CA	0	0b	When set to 1b, the PCI completion abort interrupt is enabled.
UA	1	0b	Reserved. Write 0x0, ignore on read.
BE	2	0b	When set to 1b, the wrong byte-enable interrupt is enabled.
TO	3	0b	When set to 1b, the PCI timeout interrupt is enabled.
BMEF	4	0b	When set to 1b, the BME interrupt is enabled.
ABR	5	0b	When set to 1b, the PCI completion abort received interrupt is enabled.
Reserved	31:6	0x0	Reserved. Write 0x0, ignore on read.

8.8.18 MSI-X PBA Clear - PBACL (0x5B68; RW1C)

Field	Bit(s)	Initial Value	Description
PENBITCLR	4:0	0x0	MSI-X Pending bits Clear. Writing a 1b to any bit clears the corresponding MSIXPBA bit; writing a 0b has no effect. ***Note:*** Bits are set for a single PCIe clock cycle and then cleared.
Reserved	31:5	0x0	Reserved. Write 0x0, ignore on read.

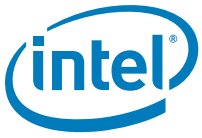
8.9 Receive Register Descriptions

8.9.1 Receive Control Register - RCTL (0x0100; RW)



Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved. Write 0b, ignore on read.
RXEN	1	0b	Receiver Enable. The receiver is enabled when this bit is set to 1b. Writing this bit to 0b stops reception after receipt of any in progress packet. All subsequent packets are then immediately dropped until this bit is set to 1b.
SBP	2	0b	Store Bad Packets 0b = Do not store. 1b = Store bad packets. This bit controls the MAC receive behavior. A packet is required to pass the address (or normal) filtering before the *SBP* bit becomes effective. If SBP = 0b, then all packets with layer 1 or 2 errors are rejected. The appropriate statistic would be incremented. If SBP = 1b, then these packets are received (and transferred to host memory). The receive descriptor error field (RDESC.ERRORS) should have the corresponding bit(s) set to signal the software device driver that the packet is erred. In some operating systems the software device driver passes this information to the protocol stack. In either case, if a packet only has layer 3+ errors, such as IP or TCP checksum errors, and passes other filters, the packet is always received (layer 3+ errors are not used as a packet filter). ***Note:*** Symbol errors before the SFD are ignored. Any packet must have a valid SFD (RX_DV with no RX_ER) in order to be recognized by Foxville (even bad packets). Also, erred packets are not routed to the MNG even if this bit is set.
UPE	3	0b	Unicast Promiscuous Enabled. 0b = Disabled. 1b = Enabled.
MPE	4	0b	Multicast Promiscuous Enabled. 0b = Disabled. 1b = Enabled.
LPE	5	0b	Long Packet Reception Enable. 0b = Disabled. 1b = Enabled. LPE controls whether long packet reception is permitted. If LPE is 0b,hardware discards long packets over 1518, 1522 or 1526 bytes depending on the *CTRL_EXT.EXT_VLAN* bit and the detection of a VLAN tag in the packet. If LPE is 1b,the maximum packet size that Foxville can receive is defined in the *RLPML.RLPML* register.
LBM	7:6	00b	Loopback Mode. Controls the loopback mode of Foxville. 00b = Nominal operation or PHY Loopback. For PHY loopback, the PHY should be set to loopback using the MDIO access.. 01b = MAC loopback. 11b, 10b = Undefined.
HSEL	9:8	00b	Hash Select for the Multicast Table Array (MTA) 00b = The MTA checks only multicast packets 01b = The MTA checks only unicast packets 10b = The MTA checks both unicast and multicast packets 11b = Reserved
Reserved	11:10	0x0	Reserved. Write 0x0, ignore on read.
MO	13:12	00b	Multicast / Unicast Table Offset. Determines which bits of the incoming multicast / unicast address are used in looking up the bit vector. 00b - bits [47:36] of received destination multicast address 01b - bits [43:32] of received destination multicast address 10b - bits [39:28] of received destination multicast address 11b - bits [35:24] of received destination multicast address

Field	Bit(s)	Initial Value	Description
Reserved	14	0b	Reserved. Write 0b, ignore on read.
BAM	15	0b	Broadcast Accept Mode. 0b = Ignore broadcast (unless it matches through exact or imperfect filters). 1b = Accept broadcast packets.
BSIZE	17:16	00b	Receive Buffer Size. BSIZE controls the size of the receive buffers and permits software to trade-off descriptor performance versus required storage space. Buffers that are 2048 bytes require only one descriptor per receive packet maximizing descriptor efficiency. 00b = 2048 Bytes. 01b = 1024 Bytes. 10b = 512 Bytes. 11b = 256 Bytes. ***Notes:*** 1. BSIZE should not be modified when RXEN is set to 1b. Set RXEN =0b when modifying the buffer size by changing this field. 2. BSIZE value only defines receive buffer size of queues with a *SRRCTL.BSIZEPACKET* value of 0b.
VFE	18	0b	VLAN Filter Enable. 0b = Disabled (filter table does not decide packet acceptance). 1b = Enabled (filter table decides packet acceptance for 802.1Q packets). Three bits [20:18] control the VLAN filter table. The first determines whether the table participates in the packet acceptance criteria. The next two are used to decide whether the CFI bit found in the 802.1Q packet should be used as part of the acceptance criteria.
CFIEN	19	0b	Canonical Form Indicator Enable. 0b = Disabled (CFI bit found in received 802.1Q packet's tag is not compared to decide packet acceptance). 1b = Enabled (CFI bit found in received 802.1Q packet's tag must match RCTL.CFI to accept 802.1Q type packet.
CFI	20	0b	Canonical Form Indicator Bit Value 0b = 802.1Q packets with CFI equal to this field are accepted. 1b = 802.1Q packet is discarded.
PSP	21	0b	Pad Small Receive Packets. If this field is set, *RCTL.SECRC* should be set.
DPF	22	1b	Discard Pause Frames. Controls whether pause frames are forwarded to the host. 0b = incoming pause frames are forwarded to the host. 1b = incoming pause frames are discarded.
PMCF	23	0b	Pass MAC Control Frames. Filters out unrecognized pause and other control frames. 0b = Filter MAC Control frames. 1b = Pass/forward MAC control frames to the Host that are not XON/XOFF flow control packets. The *PMCF* bit controls the DMA function of the MAC control frames (other than flow control). A MAC control frame in this context must be addressed to either the MAC control frame multicast address or the station address, match the type field, and NOT match the PAUSE opcode of 0x0001. If *PMCF* = 1b then frames meeting this criteria are transferred to host memory.



Field	Bit(s)	Initial Value	Description
Reserved	25:24	0x0	Reserved. Write 0x0, ignore on read.
SECRC	26	0b	Strip Ethernet CRC From Incoming Packet Causes the CRC to be stripped from all packets. 0b = Does not strip CRC. 1b = Strips CRC. This bit controls whether the hardware strips the Ethernet CRC from the received packet. This stripping occurs prior to any checksum calculations. The stripped CRC is not transferred to host memory and is not included in the length reported in the descriptor. ***Notes:*** 1. If the *CTRL.VME* bit is set the *RCTL.SECRC* bit should also be set as the CRC is not valid anymore. 2. Even when this bit is set, CRC strip is not done on runt packets (smaller than 64 bytes).
Reserved	31:27	0x0	Reserved. Write 0x0, ignore on read.

8.9.2 Packet Split Receive Type - PSRTYPE (0x5480 + 4*n [n=0...3]; RW)

This register enables or disables each type of header that needs to be split or replicated (refer to Section 7.1.5 for additional information on header split support). Each register controls the behavior of 1 queue.

— Packet Split Receive Type Register (queue 0) - *PSRTYPE0* (0x5480)

— Packet Split Receive Type Register (queue 1) - *PSRTYPE1* (0x5484)

— Packet Split Receive Type Register (queue 2) - *PSRTYPE2* (0x5488)

— Packet Split Receive Type Register (queue 3) - *PSRTYPE3* (0x548C)

Field	Bit(s)	Initial Value	Description
PSR_type0	0	0b	Header includes MAC (VLAN/SNAP).
PSR_type1	1	1b	Header includes MAC, (VLAN/SNAP) Fragmented IPv4 only.
PSR_type2	2	1b	Header includes MAC, (VLAN/SNAP) IPv4, TCP only.
PSR_type3	3	1b	Header includes MAC, (VLAN/SNAP) IPv4, UDP only.
PSR_type4	4	1b	Header includes MAC, (VLAN/SNAP) IPv4, Fragmented IPv6 only.
PSR_type5	5	1b	Header includes MAC, (VLAN/SNAP) IPv4, IPv6, TCP only.
PSR_type6	6	1b	Header includes MAC, (VLAN/SNAP) IPv4, IPv6, UDP only.
PSR_type7	7	1b	Header includes MAC, (VLAN/SNAP) Fragmented IPv6 only.
PSR_type8	8	1b	Header includes MAC, (VLAN/SNAP) IPv6, TCP only.
PSR_type9	9	1b	Header includes MAC, (VLAN/SNAP) IPv6, UDP only.
Reserved_1	10	1b	Reserved. Write 1b, ignore on read.
PSR_type11	11	1b	Header includes MAC, (VLAN/SNAP) IPv4, TCP, NFS only.
PSR_type12	12	1b	Header includes MAC, (VLAN/SNAP) IPv4, UDP, NFS only.
Reserved_1	13	1b	Reserved. Write 1b, ignore on read.
PSR_type14	14	1b	Header includes MAC, (VLAN/SNAP) IPv4, IPv6, TCP, NFS only.
PSR_type15	15	1b	Header includes MAC, (VLAN/SNAP) IPv4, IPv6, UDP, NFS only.

Field	Bit(s)	Initial Value	Description
Reserved_1	16	1b	Reserved. Write 1b, ignore on read.
PSR_type17	17	1b	Header includes MAC, (VLAN/SNAP) IPv6, TCP, NFS only.
PSR_type18	18	1b	Header includes MAC, (VLAN/SNAP) IPv6, UDP, NFS only.
Reserved	31:19	0x0	Reserved. Write 0b, ignore on read.

8.9.3 Receive Descriptor Base Address Low - RDBAL (0xC000 + 0x40*n [n=0...3]; RW)

This register contains the lower bits of the 64-bit descriptor base address. The lower four bits are always ignored. The Receive Descriptor Base Address must point to a 128 byte-aligned block of data.

Field[1]	Bit(s)	Initial Value	Description
Lower_0	6:0	0x0	Ignored on writes. Should be ignored on reads.
RDBAL	31:7	X	Receive Descriptor Base Address Low.

1. Software should program the *RDBAL[n]* register only when a queue is disabled (*RXDCTL[n].Enable* = 0b).

8.9.4 Receive Descriptor Base Address High - RDBAH (0xC004 + 0x40*n [n=0...3]; RW)

This register contains the upper 32 bits of the 64-bit descriptor base address.

Field[1]	Bit(s)	Initial Value	Description
RDBAH	31:0	X	Receive Descriptor Base Address [63:32].

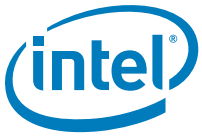
1. Software should program the *RDBAH[n]* register only when a queue is disabled (*RXDCTL[n].Enable* = 0b).

8.9.5 Receive Descriptor Ring Length - RDLEN (0xC008 + 0x40*n [n=0...3]; RW)

This register sets the number of bytes allocated for descriptors in the circular descriptor buffer. It must be 128-byte aligned.

Field[1]	Bit(s)	Initial Value	Description
Zero	6:0	0x0	Ignore on writes. Bits 6:0 must be set to 0x0. Bits 4:0 always read as 0x0.
LEN	19:7	0x0	Descriptor Ring Length (number of 8 descriptor sets). ***Note:*** Maximum allowed value in *RDLEN* field 19:0 is 0x80000 (32K descriptors).
Reserved	31:20	0x0	Reserved. Write 0x0, ignore on read.

1. Software should program the *RDLEN[n]* register only when a queue is disabled (*RXDCTL[n].Enable* = 0b).



8.9.6 Split and Replication Receive Control - SRRCTL (0xC00C + 0x40*n [n=0...3]; RW)

Field	Bit(s)	Initial Value	Description
BSIZEPACKET	6:0	0x0	Receive Buffer Size for Packet Buffer. The value is in 1 KB resolution. Valid values can be from 1 KB to 16 KB. Default buffer size is 0 KB. If this field is equal 0x0, then *RCTL.BSIZE* determines the packet buffer size.
Reserved	7	0b	Reserved
BSIZEHEADER	13:8	0x4	Receive Buffer Size for Header Buffer. The value is in 64 bytes resolution. Valid value can be from 64 bytes to 2048 bytes (BSIZEHEADER = 0x1 to 0x20). Default buffer size is 256 bytes. This field must be greater than 0 if the value of *DESCTYPE* is greater or equal to 2. ***Note:*** When *SRRCTL.Timestamp* is set to 1b and the value of *SRRCTL.DESCTYPE* is greater or equal to 2, *BSIZEHEADER* size should be equal or greater than 2 (128 bytes).
Timer1_Sel	15:14	0x0	Selects the 1588 timer reported in the Timer '0' field in the receive buffer. See Section 7.1.6 for more details.
RSVD	16	0b	
Timer0_Sel	18:17	0x0	Selects the 1588 timer reported in the Timer '1' field in the receive buffer. See Section 7.1.6 for more details.
USE_DOMAIN	19	0x0	Controls the reported timestamp in the "Timer '0'" field in the Rx descriptor: 0b - timestamp is selected according to Time0_Sel field in this register for the queue. 1b - in case of 1588 sync packets and there is a domain match, the timestamp is selected by the DOM2TIMER register, otherwise it is selected by the Timer0_Sel.
RDMTS	24:20	0x0	Receive Descriptor Minimum Threshold Size. A Low Latency Interrupt (LLI) associated with this queue is asserted each time the number of free descriptors becomes equal to *RDMTS* multiplied by 16.
DESCTYPE	27:25	000b	Defines the descriptor in Rx. 000b = Legacy. 001b = Advanced descriptor one buffer. 010b = Advanced descriptor header splitting. 011b = Advanced descriptor header replication - replicate always. 100b = Advanced descriptor header replication large packet only (larger than header buffer size). 101b = Advanced descriptor header splitting (always use header buffer). 111b = Reserved.
Reserved	29:28	0x0	Reserved. Write 0x0, ignore on read.
Timestamp	30	0b	Timestamp Received Packet 0b = Do not place timestamp at the beginning of a receive buffer. 1= Place timestamp at the beginning of a receive buffer. Timestamp is placed only in buffers of received packets that meet the criteria defined in the *TSYNCRXCTL.Type* field, 2-tuple filters or *ETQF* registers. When set, the timestamp value in *SYSTIMH* and *SYSTIML* registers is placed in the receive buffer before the MAC header of the packets defined in the *TSYNCRXCTL.Type* field.
Drop_En	31	0b/1b	Drop Enabled. If set, packets received to the queue when no descriptors are available to store them are dropped. The packet is dropped only if there are not enough free descriptors in the host descriptor ring to store the packet. If there are enough descriptors in the host, but they are not yet fetched by Foxville, then the packet is not dropped and there are no release of packets until the descriptors are fetched. Default is 0b for queue 0 and 1b for the other queues.

8.9.7 Receive Descriptor Head - RDH (0xC010 + 0x40*n [n=0...3]; RW)

The value in this register might point to descriptors that are still not in host memory. As a result, the host cannot rely on this value in order to determine which descriptor to process.

Field	Bit(s)	Initial Value	Description
RDH	15:0	0x0	Receive Descriptor Head.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.9.8 Receive Descriptor Tail - RDT (0xC018 + 0x40*n [n=0...3]; RW)

This register contains the tail pointers for the receive descriptor buffer. The register points to a 16-byte datum. Software writes the tail register to add receive descriptors to the hardware free list for the ring.

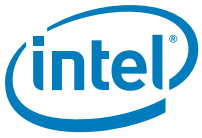
Note: Writing the RDT register while the corresponding queue is disabled is ignored by Foxville.

Field	Bit(s)	Initial Value	Description
RDT	15:0	0x0	Receive Descriptor Tail.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.9.9 Receive Descriptor Control - RXDCTL (0xC028 + 0x40*n [n=0...3]; RW)

This register controls the fetching and write-back of receive descriptors. The three threshold values are used to determine when descriptors are read from and written to host memory. The values are in units of descriptors (each descriptor is 16 bytes).

Field	Bit(s)	Initial Value	Description
PTHRESH	4:0	0xC	Prefetch Threshold PTHRESH is used to control when a prefetch of descriptors is considered. This threshold refers to the number of valid, unprocessed receive descriptors Foxville has in its on-chip buffer. If this number drops below PTHRESH, the algorithm considers pre-fetching descriptors from host memory. This fetch does not happen unless there are at least HTHRESH valid descriptors in host memory to fetch. ***Note:*** HTHRESH should be given a non zero value each time PTHRESH is used. Possible values for this field are 0 to 16.
Reserved	7:5	0x0	Reserved. Write 0x0, ignore on read.
HTHRESH	12:8	0xA	Host Threshold. This field defines when a receive descriptor prefetch is performed. Each time enough valid descriptors, as defined in the HTHRESH field, are available in host memory a prefetch is performed. Possible values for this field are 0 to 16.
Reserved	15:13	0x0	Reserved. Write 0x0, ignore on read.



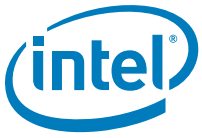
Field	Bit(s)	Initial Value	Description
WTHRESH	20:16	0x1	Write-back Threshold. *WTHRES*H controls the write-back of processed receive descriptors. This threshold refers to the number of receive descriptors in the on-chip buffer that are ready to be written back to host memory. In the absence of external events (explicit flushes), the write-back occurs only after at least *WTHRESH* descriptors are available for write-back. Possible values for this field are 0 to 15. ***Note:*** Since the default value for write-back threshold is 1b, the descriptors are normally written back as soon as one cache line is available. *WTHRESH* must contain a non-zero value to take advantage of the write-back bursting capabilities of Foxville. ***Note:*** It's recommended not to place a value above 0xC in the *WTHRESH* field.
Reserved	22:21	0x0	Reserved Write 0, ignore on read.
Reserved	23	0b	Reserved
Reserved	24	0b	Reserved Write 0, ignore on read.
ENABLE	25	0b	Receive Queue Enable. When set, the *Enable* bit enables the operation of the specific receive queue. 1b =Enables queue. 0b =Disables queue. Setting this bit initializes the Head and Tail registers (*RDH*[n] and *RDT*[n]) of the specific queue. Until then, the state of the queue is kept and can be used for debug purposes. When disabling a queue, this bit is cleared only after all activity in the queue has stopped. ***Note:*** When receive queue is enabled and descriptors exist, descriptors are fetched immediately. Actual receive activity on the port starts only if the *RCTL.RXEN* bit is set.
SWFLUSH	26	0b	Receive Software Flush. Enables software to trigger a receive descriptor write-back flushing, independently of other conditions. This bit shall be written to 1b and then to 0b after a write-back flush is triggered.
Reserved	31:27	0x0	Reserved.

8.9.10 Receive Checksum Control - RXCSUM (0x5000; RW)

The Receive Checksum Control register controls the receive checksum off loading features of Foxville. Foxville supports the off loading of three receive checksum calculations: the Packet Checksum, the IP Header Checksum, and the TCP/UDP Checksum.

Note: This register should only be initialized (written) when the receiver is not enabled (only write this register when RCTL.RXEN = 0b)

Field	Bit(s)	Initial Value	Description
PCSS	7:0	0x0	Packet Checksum Start. Controls the packet checksum calculation. The packet checksum shares the same location as the RSS field and is reported in the receive descriptor when the *RXCSUM.PCSD* bit is cleared. If the *RXCSUM.IPPCSE* is set, the Packet checksum is aimed to accelerate checksum calculation of fragmented UDP packets. Please refer to Section 7.1.7.2 for detailed explanation. If *RXCSUM.IPPCSE* is cleared (the default value), the checksum calculation that is reported in the Rx Packet checksum field is the unadjusted 16-bit ones complement of the packet. The packet checksum starts from the byte indicated by *RXCSUM.PCSS* (0b corresponds to the first byte of the packet), after VLAN stripping if enabled by the *CTRL.VME*. For example, for an Ethernet II frame encapsulated as an 802.3ac VLAN packet and with *RXCSUM.PCSS* set to 14, the packet checksum would include the entire encapsulated frame, excluding the 14-byte Ethernet header (DA, SA, Type/ Length) and the 4-byte VLAN tag. The packet checksum does not include the Ethernet CRC if the *RCTL.SECRC* bit is set. Software must make the required offsetting computation (to back out the bytes that should not have been included and to include the pseudo-header) prior to comparing the packet checksum against the L4 checksum stored in the packet checksum. The partial checksum in the descriptor is aimed to accelerate checksum calculation of fragmented UDP packets. ***Note:*** The PCSS value should point to a field that is before or equal to the IP header start. Otherwise, the IP header checksum or TCP/UDP checksum is not calculated correctly.
IPOFLD	8	1b	IP Checksum Off-load Enable. *RXCSUM.IPOFLD* is used to enable the IP Checksum off-loading feature. If *RXCSUM.IPOFLD* is set to 1b, Foxville calculates the IP checksum and indicates a pass/fail indication to software via the IP Checksum Error bit (*IPE*) in the *Error* field of the receive descriptor. Similarly, if *RXCSUM.TUOFLD* is set to 1b, Foxville calculates the TCP or UDP checksum and indicates a pass/fail indication to software via the TCP/ UDP Checksum Error bit (*RDESC.L4E*). Similarly, if *RFCTL.IPv6_DIS* and *RFCTL.IP6Xsum_DIS* are cleared to 0b and *RXCSUM.TUOFLD* is set to 1b, Foxville calculates the TCP or UDP checksum for IPv6 packets. It then indicates a pass/fail condition in the TCP/UDP *Checksum Error* bit (*RDESC.L4E*). This applies to checksum off loading only. Supported frame types: • Ethernet II • Ethernet SNAP
TUOFLD	9	1b	TCP/UDP Checksum Off-load Enable.
ICMPv6XSUM	10	1b	ICMPv6 Checksum Enable. 0b = Disable ICMPv6 checksum calculation. 1b = Enable ICMPv6 checksum calculation. ***Note:*** ICMPv6 checksum offload is supported only for packets sent to firmware for Proxying.
CRCOFL	11	0b	CRC32 Offload Enable. Enables the SCTP CRC32 checksum off-loading feature. If *RXCSUM.CRCOFL* is set to 1b, Foxville calculates the CRC32 checksum and indicates a pass/fail indication to software via the CRC32 *Checksum Valid* bit (*RDESC.L4I*) in the *Extended Status* field of the receive descriptor. In non I/OAT, this bit is read only as 0b.



Field	Bit(s)	Initial Value	Description
IPPCSE	12	0b	IP Payload Checksum Enable. See PCSS description.
PCSD	13	0b	Packet Checksum Disable. The packet checksum and IP identification fields are mutually exclusive with the RSS hash. Only one of the two options is reported in the Rx descriptor. *RXCSUM.PCSD* Legacy Rx Descriptor (*SRRCTL.DESCTYPE* = 000b): 0b (checksum enable) = Packet checksum is reported in the Rx descriptor. 1b (checksum disable) = Not supported. *RXCSUM.PCSD* Extended or Header Split Rx Descriptor (*SRRCTL.DESCTYPE* not equal 000b): 0b (checksum enable) = checksum and IP identification are reported in the Rx descriptor. 1b (checksum disable) = RSS Hash value is reported in the Rx descriptor.
Reserved	31:14	0x0	Reserved. Write 0x0, ignore on read.

8.9.11 Receive Long Packet Maximum Length - RLPML (0x5004; RW)

Field	Bit(s)	Initial Value	Description
RLPML	13:0	0x2600	Maximum allowed long packet length. This length is the global length of the packet including all the potential headers of suffixes in the packet.
Reserved	31:14	0x0	Reserved. Write 0x0, ignore on read.

8.9.12 Receive Filter Control Register - RFCTL (0x5008; RW)

Field	Bit(s)	Initial Value	Description
Reserved	5:0	1b	Reserved. Write 1b, ignore on read.
NFSW_DIS	6	0b	NFS Write Disable. Disables filtering of NFS write request headers.
NFSR_DIS	7	0b	NFS Read Disable. Disables filtering of NFS read reply headers.
NFS_VER	9:8	00b	NFS Version. 00b = NFS version 2. 01b = NFS version 3. 10b = NFS version 4. 11b = Reserved for future use.
IPv6_DIS	10	0b	IPv6 Disable Disables IPv6 packet filtering. Any received IPv6 packet is parsed only as an L2 packet.
IPv6XSUM_DIS	11	0b	IPv6 XSUM Disable. Disables XSUM on IPv6 packets.
Reserved	13:12	0x0	Reserved. Write 0x0, ignore on read.
IPFRSP_DIS	14	0b	IP Fragment Split Disable. When this bit is set, the header of IP fragmented packets are not set.

Field	Bit(s)	Initial Value	Description
Reserved	17:15	0x0	Reserved. Write 0x0 ignore on read.
LEF	18	0b	Forward Length Error Packet. 0b = Packet with length error are dropped. 1b = Packets with length error are forwarded to the host.
SYNQFP	19	0b	Defines the priority between SYNQF and 2 tuple filter. 0b = 2-tuple filter priority. 1b = SYN filter priority.

8.9.13 Multicast Table Array - MTA (0x5200 + 4*n [n=0...127]; RW)

There is one register per 32 bits of the Multicast Address Table for a total of 128 registers. Software must mask to the desired bit on reads and supply a 32-bit word on writes. The first bit of the address used to access the table is set according to the RX_CTRL.MO field.

Note: All accesses to this table must be 32 bit.

Field	Bit(s)	Initial Value	Description
Bit Vector	31:0	X	Word wide bit vector specifying 32 bits in the multicast address filter table.

Figure 8-1 shows the multicast lookup algorithm. The destination address shown represents the internally stored ordering of the received DA. Note that bit 0 indicated in this diagram is the first on the wire.

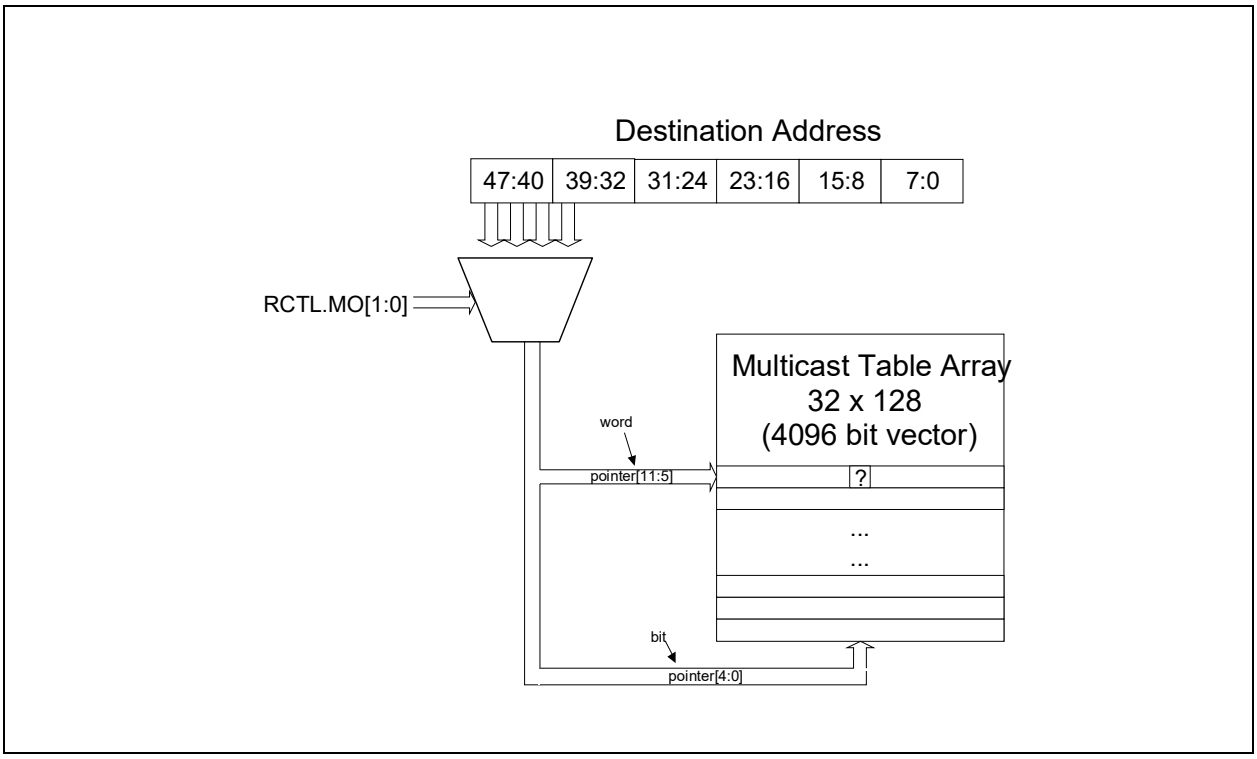


Figure 8-1. Multicast Table Array

8.9.14 Receive Address Low - RAL (0x5400 + 8*n [n=0...15]; RW)

All filters are used to filter traffic to the host. The filters are cleared by software reset and the first filter is loaded from the NVM.

Field	Bit(s)	Initial Value	Description
RAL	31:0	X	Receive Address Low. Contains the lower 32-bit of the 48-bit of the exact unicast/ multicat Ethernet MAC address filter 'n'. The LS byte in this register is the MS byte of the MAC address which is first on the wire.

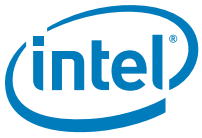
8.9.15 Receive Address High - RAH (0x5404 + 8*n [n=0...15]; RW)

See explanation of the RAL registers

Field	Bit(s)	Initial Value	Description
RAH	15:0	X	Receive address High. Contains the upper 16 bits of the 48-bit Ethernet address. The MS byte in this field is the LS byte of the MAC address which is last on the wire.
ASEL	17:16	X	Address Select. Selects how the address is to be used in the address filtering. 00b = Destination address (required for normal mode). 01b = Source address. 10b = Reserved. 11b = Reserved.
QSEL	19:18	X	Queue Select. The QSEL indicates which Rx queue should get the packets matching this MAC address. 00b = queue0. 01b = queue1. 10b = queue2. 11b = queue3.
Reserved	27:20	0x0	Reserved. Write 0x0, Ignore on reads
QSEL Enable	28	X	Queue Select Enable. When set to 1b the value in the QSEL should be used as part of the queue classification algorithm.
Reserved	30:29	0x0	Reserved. Write 0x0, ignore on reads.
AV	31	0x0	Address Valid. Cleared after master reset. If an NVM is present, the *Address Valid* field of the Receive Address Register 0 is set to 1b after a software or PCI reset or NVM read. In entries 0-15 this bit is cleared by master reset.

8.9.16 VLAN Priority Queue Filter VLANPQF (0x55B0;RW)

Field	Bit(s)	Initial value	Description
VP0QSEL	1:0	X	VLAN Priority 0 Queue Selection
VP0PBSEL	2	0b	VLAN Priority 0 Receive Packet Buffer Selection. 0b for the high priority PB and 1b for the low priority PB. This flag is meaningful only when preemption is not enabled by the PREEMPT_ENA flag in TQAVCTRL register. Note: The allocation of the high and low priority packet buffers are configured by the RXPBSIZE_EXP and RXPBSIZE_BE fields in the RXPBSIZE register.
VLANP0V	3	0b	VLAN Priority 0 Valid
VP1QSEL	5:4	X	VLAN Priority 1 Queue Selection
VP1PBSEL	6	0b	VLAN Priority 1 Rx PB Selection. See VP0PBSEL for description.
VLANP1V	7	0b	VLAN Priority 1 Valid
VP2QSEL	9:8	X	VLAN Priority 2 Queue Selection
VP2PBSEL	10	0b	VLAN Priority 2 Rx PB Selection. See VP0PBSEL for description.
VLANP2V	11	0b	VLAN Priority 2 Valid
VP3QSEL	13:12	X	VLAN Priority 3 Queue Selection



Field	Bit(s)	Initial value	Description
VP3PBSEL	14	0b	VLAN Priority 3 Rx PB Selection. See VP0PBSEL for description.
VLANP3V	15	0b	VLAN Priority 3 Valid
VP4QSEL	17:16	X	VLAN Priority 4 Queue Selection
VP4PBSEL	18	0b	VLAN Priority 4 Rx PB Selection. See VP0PBSEL for description.
VLANP4V	19	0b	VLAN Priority 4 Valid
VP5QSEL	21:20	X	VLAN Priority 5 Queue Selection
VP5PBSEL	22	0b	VLAN Priority 5 Rx PB Selection. See VP0PBSEL for description.
VLANP5V	23	0b	VLAN Priority 5 Valid
VP6QSEL	25:24	X	VLAN Priority 6 Queue Selection
VP6PBSEL	26	0b	VLAN Priority 6 Rx PB Selection. See VP0PBSEL for description.
VLANP6V	27	0b	VLAN Priority 6 Valid
VP7QSEL	29:28	X	VLAN Priority 7 Queue Selection
VP7PBSEL	30	0b	VLAN Priority 7 Rx PB Selection. See VP0PBSEL for description.
VLANP7V	31	0b	LAN Priority 7 Valid

8.9.17 VLAN Filter Table Array - VFTA (0x5600 + 4*n [n=0...127]; RW)

There is one register per 32 bits of the VLAN Filter Table. The size of the word array depends on the number of bits implemented in the VLAN Filter Table. Software must mask to the desired bit on reads and supply a 32-bit word on writes.

Note: All accesses to this table must be 32 bit.

The algorithm for VLAN filtering using the VFTA is identical to that used for the Multicast Table Array. Refer to Section 8.9.13 for a block diagram of the algorithm. If VLANs are not used, there is no need to initialize the VFTA.

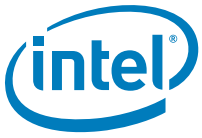
Field	Bit(s)	Initial Value	Description
Bit Vector	31:0	X	Double-word wide bit vector specifying 32 bits in the VLAN Filter table.

8.9.18 Multiple Receive Queues Command Register - MRQC (0x5818; RW)

Field	Bit(s)	Initial Value	Description
Multiple Receive Queues Enable	2:0	0x0	Multiple Receive Queues Enable. Enables support for Multiple Receive Queues and defines the mechanism that controls queue allocation. 000b = Multiple receive queues as defined by filters (2-tuple filters, L2 Ether-type filters, SYN filter and flex filters). 001b = Reserved. 010b = Multiple receive queues as defined by filters and RSS for 4 queues[1]. 011b = Reserved. 100b = Reserved. 101b = Reserved. 110b = Reserved. 111b = Reserved. Allowed values for this field are 000b, 010b. Any other value is ignored.
Def_Q	5:3	0x0	Defines the default queue according to value of the *Multiple Receive Queues Enable* field. If Multiple Receive Queues Enable equals: 000b= Def_Q defines the destination of all packets not forwarded by filters. 001b= Def_Q field is ignored 010b= Def_Q defines the destination of all packets not forwarded by RSS or filters. 011b = Def_Q field is ignored. 100-101b= Def_Q field is ignored. 110b= Def_Q field is ignored.
Reserved	15:6	0x0	Reserved. Write 0x0, ignore on read.
RSS Field Enable	31:16	0x0	Each bit, when set, enables a specific field selection to be used by the hash function. Several bits can be set at the same time. Bit[16] = Enable TcpIPv4 hash function Bit[17] = Enable IPv4 hash function Bit[18] = Enable TcpIPv6Ex hash function Bit[19] = Enable IPv6Ex hash function Bit[20] = Enable IPv6 hash function Bit[21] = Enable TCPIPv6 hash function Bit[22] = Enable UDPIPv4 Bit[23] = Enable UDPIPv6 Bit[24] = Enable UDPIPv6Ext Bit[25] = Reserved 0 Bits[31:26] = Reserved (zero).

1. Note that the *RXCSUM.PCSD* bit should be set to enable reception of the RSS hash value in the receive descriptor.

Note: The *MRQC.Multiple Receive Queues Enable* field is used to enable/disable RSS hashing and also to enable multiple receive queues. Disabling this feature is not recommended. Model usage is to reset Foxville after disabling the RSS.



8.9.19 RSS Random Key Register - RSSRK (0x5C80 + 4*n [n=0...9]; RW)

Field	Bit(s)	Initial Value	Description
K0	7:0	0x0	Byte n*4 of the RSS random key (n=0,1,...9).
K1	15:8	0x0	Byte n*4+1 of the RSS random key (n=0,1,...9).
K2	23:16	0x0	Byte n*4+2 of the RSS random key (n=0,1,...9).
K3	31:24	0x0	Byte n*4+3 of the RSS random key (n=0,1,...9).

The RSS Random Key register stores a 40 byte key used by the RSS hash function.

31 24	23 16	15 8	7 0
K[3]	K[2]	K[1]	K[0]
...	...	...	...
K[39]	...	...	K[36]

8.9.20 Redirection Table - RETA (0x5C00 + 4*n [n=0...31]; RW)

The redirection table is a 128-entry table with each entry being eight bits wide. Only 1 to 3 bits of each entry are used to store the queue index. The table is configured through the following RW registers.

Field	Bit(s)	Initial Value	Description
Entry 0	7:0	X	Determines the tag value and physical queue for index 4*n+0 (n=0...31).
Entry 1	15:8	X	Determines the tag value and physical queue for index 4*n+1 (n=0...31).
Entry 2	23:16	X	Determines the tag value and physical queue for index 4*n+2 (n=0...31).
Entry 3	31:24	X	Determines the tag value and physical queue for index 4*n+3 (n=0...31).

31 24	23 16	15 8	7 0
Tag 3	Tag 2	Tag 1	Tag 0
...	...	...	...
Tag 127	...	...	...

Each entry (byte) of the redirection table contains the following:

7:3	2:0
Reserved	Queue index

— Bits [7:3] - Reserved.

— Bits [2:0] - Queue index for all pools or in regular RSS. In RSS only mode, all bits are used.

The contents of the redirection table are not defined following reset of the Memory Configuration registers. System software must initialize the table prior to enabling multiple receive queues. It can also update the redirection table during run time. Such updates of the table are not synchronized with the arrival time of received packets. Therefore, it is not guaranteed that a table update takes effect on a specific packet boundary.

Note: In case the operating system provides a redirection table whose size is smaller than 128 bytes, the software usually replicates the operating system-provided redirection table to span the whole 128 bytes of the hardware's redirection table.

8.10 Filtering Register Descriptions

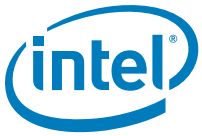
8.10.1 Immediate Interrupt RX - IMIR (0x5A80 + 4*n [n=0…7]; RW)

This IMIR[n], TTQF[n], and the IMIREXT[n] registers define the filtering required to indicate which packet triggers a LLI (immediate interrupt). The registers can also be used for queuing and deciding on the timestamp of a packet.

Notes:

1. The *Port* field should be written in network order.
2. If one of the actions for this filter is set, then at least one of the *IMIR[n].PORT_BP*, *IMIR[n].Size_BP*, the Mask bits in the TTQF[n] register or the *IMIREXT.CtrlBit_BP* bits should be cleared.
3. The value of the IMIR and IMIREXT registers after reset is unknown (apart from the *IMIR.Immediate Interrupt* bit which is guaranteed to be cleared). Therefore, both registers should be programmed before an *IMIR.Immediate Interrupt* is set for a given flow.

Field	Bit(s)	Initial Value	Description
Destination Port	15:0	0x0	Destination TCP Port This field is compared with the Destination TCP port in incoming packets. Only a packet with a matching destination TCP port triggers an immediate interrupt (if *IMIR[n].Immediate Interrupt* is set to 1b) and trigger the actions defined in the appropriate TTQF[n] register if all other filtering conditions are met. ***Note:*** Enabled by the *IMIR.PORT_BP* bit.
Immediate Interrupt	16	0b	Enables issuing an immediate interrupt when the following conditions are met: • The 2-tuple filter associated with this register matches. • The length filter associated with this filter matches. • The TCP flags filter associated with this filter matches.
PORT_BP	17	X	Port Bypass. When set to 1b, the TCP port check is bypassed and only other conditions are checked. When set to 0b, the TCP port is checked to fit the port field.
Reserved	28:18	0x0	Reserved. Write 0x0, ignore on read.
Filter Priority	31:29	000b	Defines the priority of the filter assuming two filters with same priority don't match. If two filters with the same priority match the incoming packet, the first filter (lowest ordinal number) is used in order to define the queue destination of this packet.



8.10.2 Immediate Interrupt Rx Ext. - IMIREXT (0x5AA0 + 4*n [n=0...7]; RW)

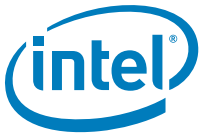
Field	Bit(s)	Initial Value	Description
Size_Thresh	11:0	X	Size Threshold. These 12 bits define a size threshold. Only a packet with a length below this threshold triggers an immediate interrupt (if *IMIR[n].Immediate Interrupt* is set to 1b) and trigger the actions defined in the appropriate TTQF[n] register (if *TTQF[n].Queue Enable* is set to 1b) if all other filtering conditions are met. ***Notes:*** 1. Enabled by the *IMIREXT.Size_BP* bit. 2. The size used for this comparison is the size of the packet as forwarded to the host and does not include any of the fields stripped by the MAC (VLAN or CRC). As a result, setting the *RCTL.SECRC* and *CTRL.VME* bits should be taken into account while calculating the size threshold. 3.
Size_BP	12	X	Size Bypass. When 1b, the size check is bypassed. When 0b, the size check is performed.
CtrlBit	18:13	X	Control Bit. Defines TCP control bits used to generate immediate interrupt and trigger filter. Only a received packet with the corresponding TCP control bits set to 1b triggers an immediate interrupt (if *IMIR[n].Immediate Interrupt* is set to 1b) and trigger the actions defined in the appropriate TTQF[n] register (if *TTQF[n].Queue Enable* is set to 1b) if all other filtering conditions are met. Bit 13 (URG)= Urgent pointer field significant. Bit 14 (ACK)= Acknowledgment field. Bit 15 (PSH):= Push function. Bit 16 (RST)= Reset the connection. Bit 17 (SYN)= Synchronize sequence numbers. Bit 18 (FIN)= No more data from sender. ***Note:*** Enabled by the *IMIREXT.CtrlBit_BP* bit.
CtrlBit_BP	19	X	Control Bits Bypass. When set to 1b, the control bits check is bypassed. When set to 0b, the control bits check is performed.
Reserved	31:20	0x0	Reserved. Write 0x0, ignore on read.

8.10.3 2-tuples Queue Filter - TTQF (0x59E0 + 4*n[n=0...7]; RW)

Field	Bit(s)	Initial Value	Description
Protocol	7:0	0x0	IP L4 protocol, part of the 2-tuple queue filters. This field is compared with the IP L4 protocol in incoming packets. Only a packet with a matching IP L4 protocol will trigger an immediate interrupt (if *IMIR[n].Immediate Interrupt* is set to 1b) and trigger the actions defined in the appropriate TTQF[n] register (if *TTQF[n].Queue Enable* is set to 1b) if all other filtering conditions are met.
Queue Enable	8	0b	When set, enables filtering of Rx packets by the 2-tuples defined in this filter to the queue indicated in this register.
Reserved	11:9	0x0	Reserved. Write 0x0, ignore on read.
Reserved	14:9	0x0	Reserved. Write 0x0, ignore on read.
Reserved_1	15	1b (For legacy reasons)	Reserved. Write 1b, ignore on read.
Rx Queue	18:16	0x0	Identifies the Rx queue associated with this 2-tuple filter. Valid values are 0 to 3.
Reserved	26:19	0x0	Reserved Write 0x0, ignore on read.
1588 time stamp	27	0b	When set, packets that match this filter are time stamped according to the IEEE 1588 specification. ***Note:*** Packet is time stamped only if it matches IEEE 1588 protocol according to the definition in the *TSYNCRXCTL.Type* field.
Mask	31:28	0xF (for legacy reasons)	Mask bits for the 2-tuple fields (the mask bit for the TCP destination port is in the *IMIR* register for legacy reasons). The corresponding field participates in the match if the following bit cleared: Bit 28 = Mask protocol comparison. Bits 31:29 = Reserved.

8.10.4 Immediate Interrupt Rx VLAN Priority - IMIRVP (0x5AC0; RW)

Field	Bit(s)	Initial Value	Description
Vlan_Pri	2:0	000b	VLAN Priority. This field includes the VLAN priority threshold. When *Vlan_pri_en* is set to 1b, then an incoming packet with a VLAN tag with a priority field equal or higher to VlanPri triggers an immediate interrupt, regardless of the EITR moderation.
Vlan_pri_en	3	0b	VLAN Priority Enable. When set to 1b, an incoming packet with VLAN tag with a priority equal or higher to Vlan_Pri triggers an immediate interrupt, regardless of the EITR moderation. When set to 0b, the interrupt is moderated by EITR.
Reserved	31:4	0x0	Reserved. Write 0x0, ignore on read.



8.10.5 SYN Packet Queue Filter - SYNQF (0x55FC; RW)

Field	Bit(s)	Initial Value	Description
Queue Enable	0	0b	When set, enables forwarding of Rx packets to the queue indicated in this register.
Rx Queue	3:1	0x0	Identifies an Rx queue associated with SYN packets. Valid values are 0 to 3.
Reserved	31:4	0x0	Reserved. Write 0x0, ignore on read.

8.10.6 EType Queue Filter - ETQF (0x5CB0 + 4*n[n=0...7]; RW)

Field	Bit(s)	Initial Value	Description
EType	15:0	0x0	Identifies the protocol running on top of IEEE 802. Used to forward Rx packets containing this EType to a specific Rx queue. The MS byte in this field is the MS byte of the Ethertype which is first on the wire.
Rx Queue	18:16	0x0	Identifies the receive queue associated with this EType. Valid values are 0 to 3.
Reserved	19	0x0	Reserved. Write 0x0, ignore on read.
EType Length	24:20	0x0	Ethertype Length. When enabled by *Ethertype length enable* this field defines the length of the Ethertype specified by EType and the device continues parsing incoming packets post this EType. The length includes the Ethertype itself as well as the data portion that is followed for this Ethertype. The minimal Ethertype length supported is 4 bytes.
EType Length Enable	25	0x0	Ethertype Length Enable. When set indicates the Ethertype length defined in EType Length is valid.
Filter enable	26	0b	When set, this filter is valid. Any of the actions controlled by the following fields are gated by this field.
Reserved	28:27	0x0	Reserved. Write 0x0, ignore on read.
Immediate Interrupt	29	0x0	When set, packets that match this filter generate an immediate interrupt.
1588 time stamp	30	0b	When set, packets with this EType are time stamped according to the IEEE 1588 specification. ***Note:*** The packet is time stamped only if it matches IEEE 1588 protocol according to the definition in the *TSYNCRXCTL.Type* field.
Queue Enable	31	0b	When set, enables filtering of Rx packets by the EType defined in this register to the queue indicated in this register.

8.11 Transmit Register Descriptions

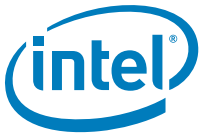
8.11.1 Transmit Control Register - TCTL (0x0400; RW)

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved. Write 0b, ignore on read.
EN	1	0b	Transmit Enable. The transmitter is enabled when this bit is set to 1b. Writing 0b to this bit stops transmission after any in progress packets are sent. Data remains in the transmit FIFO until the device is re-enabled. Software should combine this operation with reset if the packets in the TX FIFO should be flushed.
Reserved	2	0b	Reserved. Write 0b, ignore on read.
PSP	3	1b	Pad Short Packets. 0b = Do not pad. 1b = Pad. Padding makes the packet 64 bytes long. This is not the same as the minimum collision distance. If padding of short packets is allowed, the total length of a packet not including FCS should be not less than 17 bytes.
CT	11:4	0xF	Collision Threshold. This determines the number of attempts at retransmission prior to giving up on the packet (not including the first transmission attempt). While this can be varied, it should be set to a value of 15 in order to comply with the IEEE specification requiring a total of 16 attempts. The Ethernet back-off algorithm is implemented and clamps to the maximum number of slot-times after 10 retries. This field only has meaning when in half-duplex operation. ***Note:*** Software can choose to abort packet transmission in less than the Ethernet mandated 16 collisions. For this reason, hardware provides CT support.
BST	21:12	0x40	Back-Off Slot Time. This value determines the back-off slot time value in byte time. Software should initialize this field to 0x3F if HDX is used in order to meet the backoff unit of 64 bytes time.
SWXOFF	22	0b	Software XOFF Transmission. When set to 1b, Foxville schedules the transmission of an XOFF (PAUSE) frame using the current value of the PAUSE timer (*FCTTV.TTV*). This bit self-clears upon transmission of the XOFF frame. ***Note:*** While 802.3x flow control is only defined during full duplex operation, the sending of PAUSE frames via the *SWXOFF* bit is not gated by the duplex settings within Foxville. Software should not write a 1b to this bit while Foxville is configured for half-duplex operation.
Reserved	23	0b	Reserved
RTLC	24	0b	Re-transmit on Late Collision. ***Note:*** When set, enables Foxville to re-transmit on a late collision event. If a late collision is detected when this bit is disabled, the transmit function assumes the packet has successfully transmitted. This bit is ignored in full-duplex mode.
Reserved	25	0b	Reserved
Reserved	27:26	0x1	Reserved
Reserved	31:28	0xA	Reserved

8.11.2 Transmit Control Extended - TCTL_EXT (0x0404; RW)

This register controls late collision detection.

The *COLD* field is used to determine the latest time in which a collision indication is considered as a valid collision and not a late collision. When using the internal PHY, the default value of 0x40 provides a behavior consistent with the 802.3 spec requested behavior. However, when using an SGMII connected external PHY, the SGMII interface adds some delay on top of the time budget allowed by the



specification (collisions in valid network topographies even after 512 bit time can be expected). In order to accommodate this condition, *COLD* should be updated to take the SGMII inbound and outbound delays.

Field	Bit(s)	Initial Value	Description
Reserved	9:0	0x40	Reserved. Write 0x40, ignore on read.
COLD	19:10	0x42	Collision Distance. Used to determine the latest time in which a collision indication is considered as a valid collision and not a late collision.
Reserved	31:20	0x0	Reserved. Write 0x0, ignore on read.

8.11.3 Transmit IPG Register - TIPG (0x0410; RW)

This register controls the Inter Packet Gap (IPG) timer.

Field	Bit(s)	Initial Value	Description
IPGT	9:0	0x08	IPG Back to Back. Specifies the IPG length for back to back transmissions in both full and half duplex Defined in byte units. The IPG equals to IPGT + 4 while the default is 12 bytes.
IPGR1	19:10	0x04	IPG Part 1. Specifies the portion of the IPG in which the transmitter defers to receive events. IPGR1 should be set to 2/3 of the total effective IPG (8). Defined in byte units.
IPGR	29:20	0x06	IPG After Deferral. Specifies the total IPG time for non back-to-back transmissions (transmission following deferral) in half duplex. Defined in byte units. An offset of 5-byte times must be added to the programmed value to determine the total IPG after a defer event. A value of 7 is recommended to achieve a 12-byte effective IPG. Note that the IPGR must never be set to a value greater than IPGT. If IPGR is set to a value equal to or larger that IPGT, it overrides the IPGT IPG setting in half duplex resulting in inter-packet gaps that are larger then intended by IPGT. In this case, full duplex is unaffected and always relies on IPGT.
Reserved	31:30	0x0	Reserved. Write 0x0, ignore on read.

8.11.4 Retry Buffer Control - RETX_CTL (0x041C; RW)

This register controls the collision retry buffer.

Field	Bit(s)	Initial Value	Description
Water Mark	3:0	0x3	Retry buffer water mark. This parameter defines the minimal number of Qwords that should be present in the retry buffer before transmission is started.
Reserved	7:4	0x0	Reserved Write 0x0, ignore on read.

Reserved	11:8	0x0	Reserved Write 0x0, ignore on read.
Reserved	12	0x0	Reserved. Write 0x0, ignore on read.
Reserved	31:13	0x0	Reserved. Write 0x0, ignore on read.

8.11.5 DMA TX Control - DTXCTL (0x3590; RW)

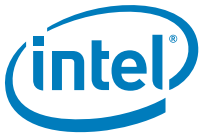
This register is used for controlling the DMA Tx behavior.

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0x0	Reserved. Write 0x0, ignore on read.
Enable_spoof_queue	2	0b	Enable Spoofing Queue. 0b = Disable queue that exhibited spoofing behavior. 1b = Do not disable port that exhibited spoofing behavior.
disable_malicious_block	3	0b	Disable Malicious Blocking. 0b = Block queue that exhibited malicious behavior on Transmit path. 1b = Do not block queue that exhibited malicious behavior on Transmit path.
OutOfSyncDisable	4	0b	Disable Out Of Sync Mechanism. 0b = Out Of Sync mechanism is enabled. 1b = Out Of Sync mechanism is disabled.
MDP_EN	5	0b	Malicious Driver Protection Enable. 0b = mechanism is disabled. 1b = mechanism is enabled.
Reserved	6	0	Reserved.
Count CRC	7	1b	If set, the CRC is counted as part of the packet bytes statistics in per Queue statistics (PQGORC, PQGOTC, PQGORLBC and PQGOTLBC).
Reserved	31:8	0x0	Reserved. Write 0x0, ignore on read.

8.11.6 DMA TX TCP Flags Control Low - DTXTCPFLGL (0x359C; RW)

This register holds the buses that AND the control flags in TCP header for the first and middle segments of a TSO packet. Refer to Section 7.2.5.7.1 and Section 7.2.5.7.2 for details on the use of this register.

Field	Bit(s)	Initial Value	Description
TCP_flg_first_seg	11:0	0xFF6	TCP Flags First Segment. Bits that are used to execute an AND operation with the TCP flags in the TCP header in the first segment
Reserved	15:12	0x0	Reserved. Write 0x0, ignore on read.
TCP_Flg_mid_seg	27:16	0xF76	TCP Flags middle segments. Bits that are used to execute an AND operation with the TCP flags in the TCP header in the middle segments.
Reserved	31:28	0x0	Reserved. Write 0x0, ignore on read.



8.11.7 DMA TX TCP Flags Control High - DTXTCPFLGH (0x35A0; RW)

This register holds the buses that AND the control flags in TCP header for the last segment of a TSO packet. Refer to Section 7.2.5.7.3 for details of use of this register.

Field	Bit(s)	Initial Value	Description
TCP_Flg_lst_seg	11:0	0xF7F	TCP Flags Last Segment. Bits that are used to execute an AND operation with the TCP flags at TCP header in the last segment.
Reserved	31:12	0x0	Reserved. Write 0x0, ignore on read.

8.11.8 DMA TX Max Total Allow Size Requests - DTXMXSZRQ (0x3540; RW)

This register limits the allowable size of concurrent outstanding Tx read requests from the host memory on the PCIe. Limiting the size of concurrent outstanding PCIe requests allows low latency packet read requests to be serviced in a timely manner, as the low latency request is serviced right after current outstanding requests are completed.

Reset by PCIe reset, LAN_PWR_GOOD and Device Reset (*CTRL.DEV_RST*) only.

Field	Bit(s)	Initial Value	Description
Max_bytes_num_req	11:0	0x10	Maximum allowable size of concurrent Tx outstanding requests on PCIe. Field defines maximum size in 256 byte resolution of outstanding Tx requests to be sent on PCIe. If total amount of outstanding Tx requests is higher than defined in this field, no further Tx outstanding requests are sent.
Reserved	30:12	0x0	Reserved Write 0, ignore on read.
Cb_enable_actual	31	0b	When bit is set, the signal lso_in_proc will not be cleared in case of force queue disable.

8.11.9 DMA TX Maximum Packet Size - DTXMXPKTSZ (0x355C; RW)

This register limits the total number of data bytes that might be transmitted in a single frame. Reducing packet size enables better utilization of transmit buffer.

Field	Bit(s)	Initial Value	Description
MAX_TPKT_SIZE	8:0	0x98	4. Maximum transmit packet size that is allowed to be transmitted by the driver. Value entered is in 64 Bytes resolution. Default value enables transmission of maximum sized 9,728-byte Jumbo frames. Values programmed in this field should not exceed 9,728 bytes. When TSN is enabled, this field should be set to 0x19 (1600 bytes).
Reserved	31:9	0x0	Reserved. Write 0x0, ignore on read.

8.11.10 Transmit Descriptor Base Address Low - TDBAL (0xE000 + 0x40*n [n=0...3]; RW)

These registers contain the lower 32 bits of the 64-bit descriptor base address. The lower 7 bits are ignored. The Transmit Descriptor Base Address must point to a 128-byte aligned block of data.

Field[1]	Bit(s)	Initial Value	Description
Lower_0	6:0	0x0	Ignored on writes. Should be ignored on reads.
TDBAL	31:7	X	Transmit Descriptor Base Address Low.

1. Software should program the TDBAL[n] register only when a queue is disabled (*TXDCTL[n].Enable* = 0b).

8.11.11 Transmit Descriptor Base Address High - TDBAH (0xE004 + 0x40*n [n=0...3]; RW)

These registers contain the upper 32 bits of the 64-bit descriptor base address.

Field[1]	Bit(s)	Initial Value	Description
TDBAH	31:0	X	Transmit Descriptor Base Address [63:32].

1. Software should program the TDBAH[n] register only when a queue is disabled (*TXDCTL[n].Enable* = 0b).

8.11.12 Transmit Descriptor Ring Length - TDLEN (0xE008 + 0x40*n [n=0...3]; RW)

These registers contain the descriptor ring length. The registers indicates the length in bytes and must be 128-byte aligned.

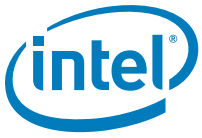
Field[1]	Bit(s)	Initial Value	Description
Zero	6:0	0x0	Ignore on writes. Read back as 0x0.
LEN	19:7	0x0	Descriptor Ring Length (number of 8 descriptor sets). ***Note:*** Maximum allowed value in *TDLEN* field 19:0 is 0x80000 (32K descriptors).
Reserved	31:20	0x0	Reserved. Write 0x0, ignore on read.

1. Software should program the TDLEN[n] register only when a queue is disabled (*TXDCTL[n].Enable* = 0b).

8.11.13 Transmit Descriptor Head - TDH (0xE010 + 0x40*n [n=0...3]; RW)

These registers contain the head pointer for the transmit descriptor ring. It points to a 16-byte datum. Hardware controls this pointer.

Note: The values in these registers might point to descriptors that are still not in host memory. As a result, the host cannot rely on these values in order to determine which descriptor to release.



Field	Bit(s)	Initial Value	Description
TDH	15:0	0x0	Transmit Descriptor Head.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.11.14 Transmit Descriptor Tail - TDT (0xE018 + 0x40*n [n=0...3]; RW)

These registers contain the tail pointer for the transmit descriptor ring and points to a 16-byte datum. Software writes the tail pointer to add more descriptors to the transmit ready queue. Hardware attempts to transmit all packets referenced by descriptors between head and tail.

Field	Bit(s)	Initial Value	Description
TDT	15:0	0x0	Transmit Descriptor Tail.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.11.15 Transmit Descriptor Control - TXDCTL (0xE028 + 0x40*n [n=0...3]; RW)

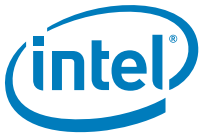
These registers control the fetching and write-back operations of transmit descriptors. The three threshold values are used to determine when descriptors are read from and written to host memory. The values are in units of descriptors (each descriptor is 16 bytes).

Since write-back of transmit descriptors is optional (under the control of *RS* bit in the descriptor), not all processed descriptors are counted with respect to *WTHRESH*. Descriptors start accumulating after a descriptor with *RS* set is processed. In addition, with transmit descriptor bursting enabled, some descriptors are written back that did not have *RS* set in their respective descriptors.

Note: When *WTHRESH* = 0x0, only descriptors with the *RS* bit set are written back.

Field	Bit(s)	Initial Value	Description
PTHRESH	4:0	0x0	Prefetch Threshold. Controls when a prefetch of descriptors is considered. This threshold refers to the number of valid, unprocessed transmit descriptors Foxville has in its on-chip buffer. If this number drops below PTHRESH, the algorithm considers pre-fetching descriptors from host memory. However, this fetch does not happen unless there are at least HTHRESH valid descriptors in host memory to fetch. ***Note:*** When PTHRESH is 0x0 a transmit descriptor fetch operation is done when any valid descriptors are available in host memory and space is available in internal buffer.
Reserved	7:5	0x0	Reserved. Write 0x0, ignore on read.
HTHRESH	12:8	0x0	Host Threshold. Prefetch of transmit descriptors is considered when number of valid transmit descriptors in host memory is at least *HTHRESH*. ***Note:*** *HTHRESH* should be given a non zero value each time PTHRESH is used.
Reserved	15:13	0x0	Reserved. Write 0x0, ignore on read.

Field	Bit(s)	Initial Value	Description
WTHRESH	20:16	0x0	Write-Back Threshold. Controls the write-back of processed transmit descriptors. This threshold refers to the number of transmit descriptors in the on-chip buffer that are ready to be written back to host memory. In the absence of external events (explicit flushes), the write-back occurs only after at least *WTHRESH* descriptors are available for write-back. Possible values for this field are 0 to 23. ***Note:*** Since the default value for write-back threshold is 0b, descriptors are normally written back as soon as they are processed. *WTHRESH* must be written to a non-zero value to take advantage of the write-back bursting capabilities of Foxville.
Reserved	22:21	0x0	Reserved Write 0, ignore on read.
Reserved	23	0b	Reserved
Reserved	24	0b	Reserved. Write 0b, ignore on read.
ENABLE	25	0b	Transmit Queue Enable. When set, this bit enables the operation of a specific transmit queue.Setting this bit initializes the Tail and Head registers (TDT[n] and TDH[n]) of a specific queue. Until then, the state of the queue is kept and can be used for debug purposes. When disabling a queue, this bit is cleared only after all transmit activity on this queue is stopped. ***Note:*** When transmit queue is enabled and descriptors exist, descriptors and data are fetched immediately. Actual transmit activity on port starts only if the *TCTL.EN* bit is set.
SWFLSH	26	0b	Transmit Software Flush. This bit enables software to trigger descriptor write-back flushing, independently of other conditions. This bit must be written to 1b and then to 0b after a write-back flush is triggered. ***Note:*** When working in head write-back mode (*TDWBAL.Head_WB_En* = 1b) *TDWBAL.WB_on_EITR* bit should be set for a transmit descriptor flush to occur.
Priority	27	0b	Transmit Queue Priority. 0b = Low priority. 1b = High priority. When set, transmit DMA resources are always allocated to the queue before low priority queues. Arbitration between transmit queues with the same priority is done in a Round Robin (RR) fashion or in most empty fashion set by the TQAVCTRL.Data_Fetch_ARB register.
HWBTHRESH	31:28	0x0	Transmit Head Write-back Threshold. If the value of field is greater than 0x0, the head write-back to host occurs only when the amount of internal pending write backs exceeds this threshold. Refer to Section 7.2.5 for additional information. ***Note:*** When activating this mode the *WB_on_EITR* bit in the TDWBAL register should be set to guarantee a write back after a timeout even if the threshold has not been reached.



8.11.16 Tx Descriptor Completion Write-Back Address Low - TDWBAL (0xE038 + 0x40*n [n=0...3]; RW)

Field[1]	Bit(s)	Initial Value	Description
Head_WB_En	0	0b	Head Write-Back Enable. 1b = Head write back is enabled. 0b = Head write back is disabled. When *head_WB_en* is set, *TXDCTL.SWFLSH* is ignored and no descriptor write back is executed.
WB_on_EITR	1	0b	When set, a head write back is done upon EITR expiration.
HeadWB_Low	31:2	0x0	Bits 31:2 of the head write-back memory location (Dword aligned). The last 2 bits of this field are ignored and are always interpreted as 00b, meaning that the actual address is Qword aligned. Bits 1:0 are always 00b.

1. Software should program the TDWBAL[n] register only when a queue is disabled (*TXDCTL[n].Enable* = 0b).

8.11.17 Tx Descriptor Completion Write-Back Address High - TDWBAH (0xE03C + 0x40*n [n=0...3];RW)

Field[1]	Bit(s)	Initial Value	Description
HeadWB_High	31:0	0x0	Highest 32 bits of the head write-back memory location.

1. Software should program the TDWBAH[n] register only when a queue is disabled (*TXDCTL[n].Enable* = 0b).

8.12 Transmit Scheduling Registers

8.12.1 Tx Qav Hi Credit - TQAVHC (0x300C+ 0x40*n [n=0...1];RW)

Field	Bit(s)	Initial Value	Description
HiCredit	31:0	0x0	Hi Credit Value. Maximum number of credits that this queue can accumulate. See Section 7.5.2.7 for a description of how this field should be calculated. Relevant only if *TransmitMode* is set to 1b (Qav).

8.12.2 Tx Qav Control TQAVCTRL (0x3570; RW)

Field	Bit(s)	Initial Value	Description
Transmit_Mode	0	0b	Transmit Mode Configuration. 0b= Legacy. 1b= TSN.
PREEMPT_ENA	1	0b	When set to 1, preemption scheme is enabled. Preemption should not be enabled before the verify / response packets are exchanged successfully with the link partner.
1588_STAT_EN	2	0b	When set to 1b, the DMA time of transmitted packets is reported in the transmit descriptors at its status write back. In this case, the TS_VAL flag is set and the DMA_TIME field is valid in the transmit descriptor write back.

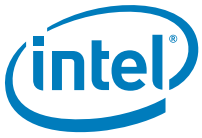
Field	Bit(s)	Initial Value	Description
Reserved	3	0b	Reserved. Software should set this bit to '1' when TSN is enabled.
Data_Fetch_ARB	4	0x0	Data Fetch Arbitration, Relevant only if Transmit_Mode is set to 1 (TSN). 0b= Round Robin. 1b= Most Empty.
Reserved	5	0x0	Reserved
Reserved	6	0x0	Reserved.
Reserved	7	0x0	Reserved
Reserved	8	0x0	Reserved
Reserved	9	0x0	Reserved
Reserved	10	0x0	Reserved
Reserved	11	0b	Reserved
Sch_Timer_Sel	13:12	0x0	Selects which of the 4 x 1588 timers is used for transmit scheduling. 00b for Timer 0 ; 01b for timer 1 ; 10b for timer 2 and 11b for timer 3.
MIN_FRAG	15:14	0x0	The minimum transmitted preempted fragment size for all non-final fragments equals to 64 * (1+MIN_FRAG) + 4 bytes of the mCRC. This is 4 byte larger than the 802.3br spec requirement. Note that the minimum size of the last transmitted fragment is 64 bytes. Note also that the minimum received preempted fragment size equals to 64 bytes.
Reserved	31:16	0x0	Reserved

8.12.3 Transmit Queue Control - TXQCTL[n] (0x3344 + 4*n, n=0...3; RW)

Reset Source: Software Reset

The parameters in this register are meaningful only in TSN mode of operation (set by the TRANSMIT_MODE field in the TQAVCTRL register). In none TSN mode all fields in this register must be set to zero.

Field	Bit(s)	Initial value	Description
QUEUE_MODE	0	0b	Transmit Queue Mode can be one of the following: 0 - Best Effort. When TQAVCTRL.TRANSMIT_MODE is set to legacy mode, the Queue Mode of all Queues must be set to BE. 1 - LaunchT: The LaunchTime in the Tx descriptor is valid. Note that transmit segmentation option is forbidden in transmit queues on which the LaunchT flag is active.
Strict_Cycle	1	0b	When set, transmission is not enabled if the whole packet cannot be completed before the end of the Qbv cycle. See note [1] below.
Strict_End	2	0b	When set, transmission is not enabled if the whole packet cannot be completed before the end of the Qbv window. See note [1] below.
Preemptable	3	0b	Queue Setting option of preemptive (Express) vs. preemptable (non-Express). 0b - Preemptive Queue (eTXQ) 1b - Preemptable Queue (pTXQ) When preemption is not enabled, all queues should be set as preemptive.
Report_Timer_Sel	5:4	00b	Selects which of the 4 timers should be reported in the DMA time stamp of the queue. See Section 7.2.2.3.1 for details.



Field	Bit(s)	Initial value	Description
Qav_Sel	7:6	00b	Enable Qav credit based shaping to the transmit queue: 00b - Credit based shaping is not enabled to the transmit queue 01b - Reserved 10b - Assign credit based shaping set 0 of the credit based registers 11b - Assign credit based shaping set 1 of the credit based registers
RESERVED	14:8	0x0	Reserved
Data_Fetch_TIM	15	0b	Enables the Fetch_Tim_Delta parameter of the queue. The Data_Fetch_TIM must be cleared for BE queues.
Fetch_Tim_Delta	31:16	0x0	Fetch Time Delta. This field holds the value to be reduced from the launch time for fetch time decision defined in 1 ns units. The Fetch_Tim_Delta must be cleared if the Data_Fetch_TIM flag in this register is cleared.

1. At 1Gb/s and at 100Mb/s the device calculates the duration of the transmitted packets accurately. However at 2.5Gb/s the device approximate the packet's transmission time as ~2.5/2.46 * the actual transmission time. If the approximate time is insufficient for the packet transmission, then the packet is not transmitted.

8.12.4 Tx Qav Credit Control TQAVCC[n] (0x3004, 0x3044;RW)

Field	Bit(s)	Initial Value	Description
IDLE_SLOPE	15:0	0x0	IdleSlope - Idle Slope for this queue Value in credits/(2 bytes) Must be smaller than LinkRate = 0x7736 credits/byte Relevant only if TQAVCTRL.TRANSMIT_MODE is set to Qav
RESERVED	29:16	0x0	
KEEP_CREDITS	30	0b	When Keep_Creidts is set and there are valid descriptors to be processed for the queue the queue credits will not be cleared even if there is no packet to send in the packet buffer.
Reserved	31	0b	Reserved

8.12.5 Tx Arbitration Control TxARB (0x3354;RW)

Field	Bit(s)	Initial Value	Description
TxQ_Priority_0	1:0	00b	The transmit queue that is assigned as priority 0 (highest priority). Default is queue 0.
TxQ_Priority_1	3:2	01b	The transmit queue that is assigned as priority 1. Default is queue 1.
TxQ_Priority_2	5:4	10b	The transmit queue that is assigned as priority 2. Default is queue 2.
TxQ_Priority_3	7:6	11b	The transmit queue that is assigned as priority 3 (lowest priority). Default is queue 3.
RSVD	15:8	0x0	Reserved
MNG_Arb	17:16	11b	The transmit queue that shares its arbitration with the Manageability traffic. Between the two, the Manageability traffic has priority over the host queue. It is meaningful only when the Transmit_Mode flag in the TQAVCTRL register is set to Qav and it must be set to zero otherwise.
RSVD	31:18	0x0	Reserved

8.12.6 Transmit Scheduling Offset - GTxOFFSET (0x3310; RW)

Reset Source: Software Reset

Field	Bit(s)	Initial Value	Description
TxOffset	15:0	0b	Global transmit offset defined in nsec units. The global offset should be set to the latency between the packet scheduling point and the network pins.
Reserved	31:16	0x0	Reserved

8.12.7 Scheduling Base Time Register Low - BASET_L (0x3314; RW)

Reset Source: Software Reset

Field	Bit(s)	Initial Value	Description
Base_Time_L	29:0	0x0	QAV Base time Low (nsec units). The Launch time in the transmit descriptor is defined relative to this base register.
Auto_Set	30	0b	Setting the Auto_Set flag, the BaseT gets the "closest" value to SYSTIM. See "Autonomous Setting option" paragraph in Section 7.5.2.9.3.2 for a description of its functionality. This flag is auto cleared by the device. Note: Setting the Auto_Set the Copy_H flag must be set as well.
Copy_H	31	0b	Setting this flag, the high portion of the register is taken from the high portion of the SYSTIM register (the one that is selected for scheduling by the TQAVCTRL.Sch_Timer_Sel). This flag is auto cleared by the device. Note: Setting the Copy_H the Auto_Set flag must be set as well.

8.12.8 Scheduling Base Time Register High - BASET_H (0x3318; RW)

Reset Source: Software Reset

Field	Bit(s)	Initial Value	Description
Base_Time_H	31:0	0x0	QAV Base time High (sec units). The Launch time in the transmit descriptor is defined relative to this base register.

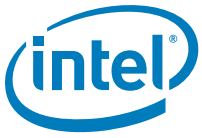
8.12.9 Qbv Cycle Time - QbvCycleT (0x331C; RW)

Reset Source: Software Reset

Field	Bit(s)	Initial Value	Description
Qbv_Cycle_Time	29:0	0x3B9ACA00	This field defines the QBV cycle time in nsec units. It is default to 1 sec and it must always be larger or equal than 31.25 usec. See Table 7-63 for permitted values.
Reserved	31:30	0x0	Reserved

8.12.10 Shadow Qbv Cycle Time - QbvCycleT_S (0x3320; RW)

Reset Source: Software Reset



Field	Bit(s)	Initial Value	Description
Shadow_QbvCycle	29:0	0x0	This field defines an internal shadow value of the QBV cycle time in nsec units. It must be set to zero when the TSN_Mode = "Basic Scheduling" and must be greater than zero when the TSN_Mode = "Qbv Scheduling". See Table 7-63 for permitted values.
Reserved	30	0x0	Reserved
Plus1	31	0x0	The Plus1 flag controls the Auto_Set option selected in the BASET_L register. See "Autonomous Setting option" paragraph in Section 7.5.2.9.3.2 for a description of its functionality.

8.12.11 Qbv Start Time - StQT (0x3324 + 0x4*n [n=0...3]; RW)

Reset Source: Software Reset

Field	Bit(s)	Initial Value	Description
Start_Time	29:0	0x0	Qbv Start time (nsec units) relative to BaseT.
RSVD	31:30	0x0	Reserved

8.12.12 Qbv End Time - EndQT (0x3334 + 0x4*n [n=0...3]; RW)

Reset Source: Software Reset

Field	Bit(s)	Initial Value	Description
END_Time	29:0	0x3B9ACA00	Qbv End time (nsec units) relative to BaseT. The END_Time minus Start_Time of the queue must be larger than the maximum between the equivalent transmission time of: 512 bytes and the maximum packet size expected in this queue plus 10%. Default setting is whole 1 sec.
RSVD	31:30	0x0	Reserved

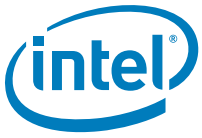
8.13 DCA and TPH Register Descriptions

8.13.1 Rx DCA Control Registers - RXCTL (0xC014 + 0x40*n [n=0...3]; RW)

Note: Rx data write no-snoop is activated when the *NSE* bit is set in the receive descriptor.

Note: Both the *DCA Enable* bit and *TPH Enable* bit should not be set for the same type of traffic.

Field	Bit(s)	Initial Value	Description
Rx Descriptor Fetch TPH EN	0	0b	Receive Descriptor Fetch TPH Enable. When set, hardware enables TPH for all Rx descriptors fetch from memory. When cleared, hardware does not enable TPH for descriptor fetches. This bit is cleared as a default.
Rx Descriptor Writeback TPH EN	1	0b	Receive Descriptor Writeback TPH Enable. When set, hardware enables TPH for all Rx descriptors written back into memory. When cleared, hardware does not enable TPH for descriptor write-backs. This bit is cleared as a default. The hint used is the hint set in the *Socket ID* field.
Rx Header TPH EN	2	0b	Receive Header TPH Enable. When set, hardware enables TPH for all received header buffers. When cleared, hardware does not enable TPH for Rx headers. This bit is cleared as a default. The hint used is the hint set in the *Socket ID* field.
Rx Payload TPH EN	3	0b	Receive Payload TPH Enable. When set, hardware enables TPH for all Ethernet payloads written into memory. When cleared, hardware does not enable TPH for Ethernet payloads. This bit is cleared as a default. The hint used is the hint set in the *Socket ID* field.
Reserved	4	0b	Reserved. Write 0b, ignore on read.
Rx Descriptor DCA EN	5	0b	Descriptor DCA Enable. When set, hardware enables DCA for all Rx descriptors written back into memory. When cleared, hardware does not enable DCA for descriptor write-backs. This bit is cleared as a default.
Rx Header DCA EN	6	0b	Receive Header DCA Enable. When set, hardware enables DCA for all received header buffers. When cleared, hardware does not enable DCA for Rx headers. This bit is cleared as a default.
Rx Payload DCA EN	7	0b	Receive Payload DCA Enable. When set, hardware enables DCA for all Ethernet payloads written into memory. When cleared, hardware does not enable DCA for Ethernet payloads. This bit is cleared as a default.
RXdescRead NSEn	8	0b	Receive Descriptor Read No Snoop Enable. This bit must be reset to 0b to ensure correct functionality (except if the software driver can guarantee the data is present in the main memory before the DMA process occurs). ***Note:*** When TPH is enabled, the *No Snoop* bit should be 0b.
RXdescRead ROEn	9	1b	Receive Descriptor Read Relax Order Enable.
RXdescWBNSen	10	0b	Receive Descriptor Write-Back No Snoop Enable. This bit must be reset to 0b to ensure correct functionality of descriptor write back. ***Note:*** When TPH is enabled *No Snoop* bit should be 0b.
RXdescWBROen (RO)	11	0b	Receive Descriptor Write-Back Relax Order Enable. This bit must be reset to 0b to ensure correct functionality of descriptor write back.
RXdataWrite NSEn	12	0b	Receive Data Write No Snoop Enable (header replication: header and data). When set to 0b, the last bit of the *Packet Buffer Address* field in the advanced receive descriptor is used as the LSB of the packet buffer address (A0), thus enabling Byte alignment of the buffer. When set to 1b, the last bit of the *Packet Buffer Address* field in advanced receive descriptor is used as the No-Snoop Enabling (NSE) bit (buffer is Word aligned). If also set to 1b, the NSE bit determines whether the data buffer is snooped or not. ***Note:*** When TPH is enabled *No Snoop* bit should be 0b.
RXdataWrite ROEn	13	1b	Receive Data Write Relax Order Enable (header replication: header and data).
RxRepHeader NSEn	14	0b	Receive Replicated/Split Header No Snoop Enable. This bit must be reset to 0b to ensure correct functionality of header write to host memory. ***Note:*** When TPH is enabled, the *No Snoop* bit should be 0b.



Field	Bit(s)	Initial Value	Description
RxRepHeader ROEn	15	1b	Receive Replicated/Split Header Relax Order Enable.
Reserved	19:16	0x0	Reserved. Write 0x0, ignore on read.
Reserved	22:20	0x0	Reserved
Reserved	23	0b	Reserved
Reserved	31:24	0x0	Reserved

8.13.2 Tx DCA Control Registers - TXCTL (0xE014 + 0x40*n [n=0...3]; RW)

Field	Bit(s)	Initial Value	Description
Tx Descriptor Fetch TPH EN[1]	0	0b	Transmit Descriptor Fetch TPH Enable. When set, hardware enables TPH for all Tx descriptors fetch from memory. When cleared, hardware does not enable TPH for descriptor fetches. This bit is cleared as a default.
Tx Descriptor Writeback TPH EN	1	0b	Transmit Descriptor Writeback TPH Enable. When set, hardware enables TPH for all Tx descriptors written back into memory. When cleared, hardware does not enable TPH for descriptor write-backs. This bit is cleared as a default. The hint used is the hint set in the *Socket ID* field.
Reserved	2	0b	Reserved. Write 0b, ignore on read.
Tx Packet TPH EN	3	0b	Transmit Packet TPH Enable. When set, hardware enables TPH for all Ethernet payloads read from memory. When cleared, hardware does not enable TPH for Ethernet payloads. This bit is cleared as a default.
Reserved	4	0b	Reserved. Write 0b, ignore on read.
Tx Descriptor DCA EN[1]	5	0b	Descriptor DCA Enable. When set, hardware enables DCA for all Tx descriptors written back into memory. When cleared, hardware does not enable DCA for descriptor write backs. This bit is cleared as a default and also applies to head write back when enabled.
Reserved	7:6	00b	Reserved. Write 00b, ignore on read.
TXdescRDNSen	8	0b	Tx Descriptor Read No Snoop Enable. This bit must be reset to 0b to ensure correct functionality (unless the software device driver has written this bit with a write-through instruction). ***Note:*** When TPH is enabled *No Snoop* bit should be 0b.
TXdescRDROEn	9	1b	Tx Descriptor Read Relax Order Enable.
TXdescWBNSen	10	0b	Tx Descriptor Write-Back No Snoop Enable. This bit must be reset to 0b to ensure correct functionality of descriptor write-back. Also applies to head write-back, when enabled. ***Note:*** When TPH is enabled *No Snoop* bit should be 0b.
TXdescWBROen	11	1b	Tx Descriptor Write Back Relax Order Enable. Applies to head write back, when enabled.
TXDataReadNSEn	12	0b	Tx Data Read No Snoop Enable. ***Note:*** When TPH is enabled *No Snoop* bit should be 0b.
TXDataReadROEn	13	1b	Tx Data Read Relax Order Enable.
Reserved	19:14	0b	Reserved Write 0 ignore on read.

Field	Bit(s)	Initial Value	Description
Reserved	22:20	0x0	Reserved
Reserved	23	0b	Reserved
Reserved	31:24	0x0	Reserved

1. Both the *DCA Enable* bit and the *TPH Enable* bit should not be set for the same type of traffic.

8.14 Timer Registers Description

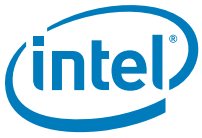
8.14.1 Watchdog Setup - WDSTP (0x1040; RW)

Field	Bit(s)	Initial Value	Description
WD_Enable	0	0b[1]	Enable Watchdog Timer.
WD_Timer_ Load_enable (SC)	1	0b	Enables the load of the watchdog timer by writing to *WD_Timer* field. If this bit is not set, the *WD_Timer* field is loaded by the value of *WD_Timeout*. ***Note:*** Writing to this field is only for DFX purposes. This field resets on software reset events.
Reserved	15:2	0x0	Reserved. Write 0x0, ignore on read.
WD_Timer (RWM)	23:16	WD_Timeout	Indicates the current value of the timer. Resets to the timeout value each time Foxville functional bit in Software Device Status register is set. If this timer expires, the WD interrupt to the firmware and the WD SDP is asserted. As a result, this timer is stuck at zero until it is re-armed. ***Note:*** Writing to this field is only for DFX purposes.
WD_Timeout	31:24	0x2[1]	Defines the number of seconds until the watchdog expires. The granularity of this timer is 1 second. The minimal value allowed for this register when the watchdog mechanism is enabled is two. Setting this field to 1b might cause the watchdog to expire immediately. ***Note:*** Only 4 LSB bits loaded from NVM.

1. Value read from the NVM.

8.14.2 Watchdog Software Device Status - WDSWSTS (0x1044; RW)

Field	Bit(s)	Initial Value	Description
Dev_Functional (SC)	0	0b	Each time this bit is set, the watchdog timer is re-armed. This bit is self clearing.
Force_WD (SC)	1	0b	Setting this bit causes the WD timer to expire immediately. The *WD_timer* field is set to 0b. It can be used by software in order to indicate some fatal error detected in the software or in the hardware. This bit is self clearing.
Reserved	23:2	0x0	Reserved. Write 0x0, ignore on read.
Stuck Reason	31:24	0x0	This field can be used by software to indicate to the firmware the reason Foxville is malfunctioning. The encoding of this field is software/firmware dependent. A value of 0x0 indicates a functional Foxville.



8.14.3 Free Running Timer - FRTIMER (0x1048; RWM)

This register reflects the value of a free running timer that can be used for various timeout indications. The register is reset by a PCI reset and/or software reset.

Note: Writing to this register is for DFX purposes only.

Field	Bit(s)	Initial Value	Description
Microsecond	9:0	X	Number of microseconds in the current millisecond.
Millisecond	19:10	X	Number of milliseconds in the current second.
Seconds	31:20	X	Number of seconds from the timer start (up to 4095 seconds).

8.14.4 TCP Timer - TCPTIMER (0x104C; RW)

Field	Bit(s)	Initial Value	Description
Duration	7:0	0x0	Duration. Duration of the TCP interrupt interval in ms.
KickStart (WO)	8	0b	Counter KickStart. Writing a 1b to this bit kick starts the counter down count from the initial value defined in the *Duration* field. Writing a 0b has no effect.
TCPCountEn	9	0b	TCP Count Enable. 1b = TCP timer counting enabled. 0b = TCP timer counting disabled. Once enabled, the TCP counter counts from its internal state. If the internal state is equal to 0b, the down-count does not restart until *KickStart* is activated. If the internal state is not 0b, the down-count continues from internal state. This enables a pause in the counting for debug purpose.
TCPCountFinish (WO)	10	0b	TCP Count Finish. This bit enables software to trigger a TCP timer interrupt, regardless of the internal state. Writing a 1b to this bit triggers an interrupt and resets the internal counter to its initial value. Down count does not restart until either *KickStart* is activated or *Loop* is set. Writing a 0b has no effect.
Loop	11	0b	TCP Loop. When set to 1b, the TCP counter reloads duration each time it reaches zero, and continues down-counting from this point without kick starting. When set to 0b, the TCP counter stops at a zero value and does not re-start until *KickStart* is activated. ***Note:*** Setting this bit alone is not enough to start the timer activity. The *KickStart* bit should also be set.
Reserved	31:12	0x0	Reserved. Write 0x0, ignore on read.

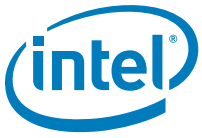
8.15 Time Sync Register Descriptions

8.15.1 Rx Time Sync Control Register - TSYNCRXCTL (0xB620;RW)

Field	Bit(s)	Initial Value	Description
RSVD	0	0x0	Reserved
Type	3:1	0x0	Type of Packets to Timestamp. 000b = Timestamp L2 (V2) packets with *MessageType* as defined by *MSGT* field in the *TSYNCRXCFG* register as well as DELAY_REQ and DELAY_RESP packets. 001b = Timestamp L4 (V1) packets with *Control* as defined by *CTRLT* field in the *TSYNCRXCFG* register. 010b = Timestamp V2 (L2 and L4) packets with *MessageType* as defined by *MSGT* field in the *TSYNCRXCFG* register as well as DELAY_REQ and DELAY_RESP packets. 100b = timestamp all packets. 101b = Timestamp all V2 packets which have a MessageType bit 3 zero, which means timestamp all event packets. 011b, 110b and 111b = Reserved
En	4	0b	Enable Rx Timestamp 0b = Timestamping disabled. 1b = Timestamping enabled.
RSVD	5	0b	Reserved
RSVD	9:6	0x0	Reserved
RSVD High	10	0x0	Reserved. **Must be set to '1' by software**. (note that setting this bit to 1b its value is reflected on bit #9)
RSV	31:11	0x0	Reserved. Write 0x0, ignore on read.

8.15.2 Tx Time Sync Control Register - TSYNCTXCTL (0xB614; RW)

Field	Bit(s)	Initial Value	Description
TXTT_0 (ROM)	0	0b	Transmit timestamp valid (equals 1b when a valid value for Tx timestamp is captured in the Tx timestamp 0 register, clear by read of Tx timestamp 0 register TXSTMPH0).
TXTT_1 (ROM)	1	0b	Transmit timestamp valid (equals 1b when a valid value for Tx timestamp is captured in the Tx timestamp 1 register, clear by read of Tx timestamp 1 register TXSTMPH1).
TXTT_2 (ROM)	2	0b	Transmit timestamp valid (equals 1b when a valid value for Tx timestamp is captured in the Tx timestamp 2 register, clear by read of Tx timestamp 2 register TXSTMPH2).
TXTT_3 (ROM)	3	0b	Transmit timestamp valid (equals 1b when a valid value for Tx timestamp is captured in the Tx timestamp 3 register, clear by read of Tx timestamp 3 register TXSTMPH3).
EN	4	0b	Enable Transmit timestamp by the TXSTMP registers. 0b = time stamping disabled. 1b = time stamping enabled.
RSVD High	5	0b	Reserved. **Must be set to '1' by software**.
Reserved	7:6	0x0	Reserved
1588_Offset	8:15	0x0	Byte offset of the inserted timestamp to the transmit packet in 1-step flow. The offset is defined in byte units measured from the beginning of the packet as transmitted to the network (including the optional inserted VLAN tag).
RSV	31:16	0x0	Reserved. Write 0x0, ignore on read.



8.15.3 Rx Time Sync Control Register - DOM2TIMER (0xB678; RW)

Field	Bit(s)	Initial Value	Description
DOM2T0	7:0	0x0	The 1588 Domain that should be reported with 1588 timer 0.
DOM2T1	15:8	0x0	The 1588 Domain that should be reported with 1588 timer 1.
DOM2T2	23:16	0b	The 1588 Domain that should be reported with 1588 timer 2.
DOM2T3	31:24	0b	The 1588 Domain that should be reported with 1588 timer 3.

8.15.4 Tx Timestamp Value Low 0 - TXSTMPL_0 (0xB618;RO)

TXSTMPL_1, TXSTMPL_2, TXSTMPL_3 are time stamps registers 1, 2, 3 respectively and have the same structure as TXSTMPL_0.

Field	Bit(s)	Initial Value	Description
TTSL	29:0	0x0	Transmit timestamp LSB value (defined in ns units).
Zero	31:30	0x0	Zero bits.

8.15.5 Tx Timestamp 0 Value High - TXSTMPH_0 (0xB61C; RO)

TXSTMPH_1, TXSTMPH_2, TXSTMPH_3 are time stamps registers 1, 2, 3 respectively and have the same structure as TXSTMPH_0.

Field	Bit(s)	Initial Value	Description
TTSH	31:0	0x0	Transmit timestamp MSB value (defined in sec units).

8.15.6 System Time Register Residue 0 - SYSTIMR_0 (0xB6F8; RW)

Field	Bit(s)	Initial Value	Description
STR	31:0	0x0	System time Residue value (defined in 2^{-32} nS resolution).

8.15.7 System Time Register Low 0 - SYSTIML_0 (0xB600; RW)

Field	Bit(s)	Initial Value	Description
STL	29:0	0x0	System time LSB value (defined in ns units).
Zero	31:30	0x0	Zero bits.

8.15.8 System Time Register High 0 - SYSTIMH_0 (0xB604; RW)

Field	Bit(s)	Initial Value	Description
STH	31:0	0x0	System time MSB value (defined in sec units).

8.15.9 System Time Register Tx MS 0 - SYSTIMTM_0 (0xB6FC; RW)

Field	Bit(s)	Initial Value	Description
STM	15:0	0x0	Two MS bytes of the system time (defined in 2^{32} sec units). This field is static, kept at the value programmed by the software. It is used for 1-step transmission as the two MS bytes inserted to the SYNC packet.
RSV	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.15.10 Increment Attributes Register 0 - TIMINCA_0 (0xB608; RW)

Field	Bit(s)	Initial Value	Description
Incvalue	30:0	0x0	Increment value. This register adjust the SYSTIM step value so it is defined in nsec units (see Section 7.5.1.3.1 for description). The Incvalue defines the value to be added or subtracted from the hard-coded step value (depending on the ISGN bit).
ISGN	31	0b	Increment sign. 0b = The Incvalue in this register is added to the hard-coded step value. 1b = The hard-coded step value is subtracted by the Incvalue.

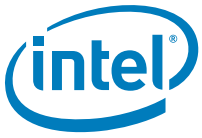
8.15.11 Time Adjustment Offset Register 0 - TIMADJ_0 (0xB60C; RW)

Field	Bit(s)	Initial Value	Description
Tadjust	29:0	0x0	Time Adjustment Value. Low (defined in ns units). The TADJL field can be set to any non-zero value smaller than 999,999,900 decimal (slightly below 1 second).
One	30	0b	Reserved One: at programming software must set this bit to 1b.
Sign	31	0b	Sign (0b= “+”1b = “-”).

8.15.12 System Time Registers for Timers 1 ... 3

The following XXX_1, XXX_2 and XXX_3 registers are the 1588 timer registers for timers 1, 2 and 3 respectively. Their structure and functionality is the same as the matched XXX_0 registers defined for the 1588 timer 0. Listed below are these registers:

- SYSTIMR_1, SYSTIMR_2, SYSTIMR_3



- SYSTIML_1, SYSTIML_2, SYSTIML_3
- SYSTIMH_1, SYSTIMH_2, SYSTIMH_3
- SYSTIMTM_1, SYSTIMTM_2, SYSTIMTM_3
- TIMINCA_1, TIMINCA_2, TIMINCA_3
- TIMADJ_1, TIMADJ_2, TIMADJ_3

8.15.13 TimeSync Auxiliary Control Register - TSAUXC (0xB640; RW)

Field	Bit(s)	Initial Value	Description
EN_TT0	0	0b	Enable target time 0. Enable bit is set by software to 1b, to enable pulse or level change generation as a function of the *TSAUXC.PLSG* bit. The bit is auto-cleared by hardware when the target time hits if the DIS_TS_CLEAR flag in this register is cleared. During nominal operation the DIS_TS_CLEAR flag is set and the EN_TT0 is not auto cleared by the hardware.
EN_TT1	1	0b	Enable target time 1. Enable bit is set by software to 1b, to enable a level change. The bit is auto-cleared by hardware when the target time hits if the DIS_TS_CLEAR flag in this register is cleared. During nominal operation the DIS_TS_CLEAR flag is set and the EN_TT0 is not auto cleared by the hardware.
EN_CLK0	2	0b	Enable Configurable Frequency Clock 0. Clock is generated according to frequency defined in the FREQOUT0 register on the SDP pin (0 to 3) that has both: 1. *TSSDP.TS_SDPx_SEL* field with a value of 10b. 2. *TSSDP.TS_SDPx_EN* value of 1b.
Reserved	4	0b	Reserved.
EN_CLK1	5	0b	Enable Configurable Frequency Clock 1. Clock is generated according to frequency defined in the FREQOUT1 register on the SDP pin (0 to 3) that has both: 1. *TSSDP.TS_SDPx_SEL* field with a value of 11b. 2. *TSSDP.TS_SDPx_EN* value of 1b.
Reserved	7	0b	Reserved.
EN_TS0	8	0b	Enable hardware timestamp 0. Enable Timestamping occurrence of change in SDP pin into the *AUXSTMPL0* and *AUXSTMPH0* registers. SDP pin (0 to 3) is selected for time stamping, if the SDP pin is selected via the *TSSDP.AUX0_SDP_SEL* field and the *TSSDP.AUX0_TS_SDP_EN* bit is set to 1b.
AUTT0	9	0b	Auxiliary Timestamp Taken. Cleared when read from auxiliary timestamp 0 occurred.
EN_TS1	10	0b	Enable Hardware Timestamp 1. Enable timestamping occurrence of change in SDP pin into the AUXSTMPL1 and AUXSTMPH1 registers. SDP pin (0 to 3) is selected for time stamping, if the SDP pin is selected via the *TSSDP.AUX1_SDP_SEL* field and the *TSSDP.AUX1_TS_SDP_EN* bit is set to 1b.
AUTT1	11	0b	Auxiliary Timestamp Taken. Cleared when read from auxiliary timestamp 1 occurred.
Reserved	16:12	0x0	Reserved. Write 0x0, ignore on read.

Field	Bit(s)	Initial Value	Description
PLSG	17	0b	Use Target Time 0 to generate start of pulse and Target Time 1 to generate end of pulse. SDP pin selected to drive pulse or level change is set according to the *TSSDP.TS_SDPx_SEL* field with a value of 00b and *TSSDP.TS_SDPx_EN* bit with a value of 1b. 0b = Target Time 0 generates change in SDP level. 1b = Target time 0 generates start of pulse on SDP pin. ***Note:*** Pulse or level change is generated when *TSAUXC.EN_TT0* is set to 1b.
Reserved	26:18	0b	Reserved. Write 0b, ignore on read.
Disable systime 1	27	1b	Disable SYSTIM 1 Count Operation. 0b = SYSTIM timer activated 1b = *SYSTIM* timer disabled. Value of *SYSTIM registers* remain constant.
Disable systime 2	28	1b	Disable SYSTIM 2 Count Operation. 0b = SYSTIM timer activated 1b = *SYSTIM* timer disabled. Value of *SYSTIM registers* remain constant.
Disable systime 3	29	1b	Disable SYSTIM 3 Count Operation. 0b = SYSTIM timer activated 1b = *SYSTIM* timer disabled. Value of *SYSTIM registers* remain constant.
DIS_TS_CLEAR	30	1b	When set to 1b the EN_TT0 and EN_TT1 flags in this register is not auto-cleared by the hardware. As a results, the CLK0 and CLK1 are able to generate interrupts.
Disable systime 0	31	1b	Disable SYSTIM 0 Count Operation. 0b = SYSTIM timer activated 1b = *SYSTIM* timer disabled. Value of *SYSTIM registers* remain constant.

8.15.14 Target Time Register 0 Low - TRGTTIML0 (0xB644; RW)

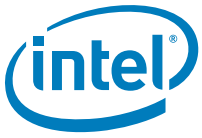
Field	Bit(s)	Initial Value	Description
TTL	29:0	0x0	Target Time 0 LSB register (defined in ns units).
IO_TIMER_SEL	31:30	0x0	Select one of the 4 1588 timers that the target time is associated with.

8.15.15 Target Time Register 0 High - TRGTTIMH0 (0xB648; RW)

Field	Bit(s)	Initial Value	Description
TTH	31:0	0x0	Target Time 0 MSB register (defined in second units).

8.15.16 Target Time Register 1 Low - TRGTTIML1 (0xB64C; RW)

Field	Bit(s)	Initial Value	Description
TTL	29:0	0x0	Target Time 1 LSB register (defined in ns units).
IO_TIMER_SEL	31:30	0x0	Select one of the 4 1588 timers that the target time is associated with.



8.15.17 Target Time Register 1 High - TRGTTIMH1 (0xB650; RW)

Field	Bit(s)	Initial Value	Description
TTH	31:0	0x0	Target Time 1 MSB register (defined in second units).

8.15.18 Frequency Out 0 Control Register FREQOUT0 (0xB654; RW)

Field	Bit(s)	Initial Value	Description
CHCT	29:0	0x0	Clock Out Half Cycle Time defines the Half Cycle time of Clock 0 in nsec units. When clock output is enabled, permitted values are any value larger than 8 and up to including 70,000,000 decimal (70 ms). The following larger values can be used as long as the output clock is synchronized to whole seconds as described in section "Synchronized Output Clock on SDP Pins": 125 ms; 250 ms and 500 ms.
Reserved	31:30	0x0	Reserved. Write 0x0, ignore on read.

8.15.19 Frequency Out 1 Control Register - FREQOUT1 (0xB658; RW)

Field	Bit(s)	Initial Value	Description
CHCT	29:0	0x0	Clock Out Half Cycle Time defines the Half Cycle time of Clock 1 in nsec units. When clock output is enabled, permitted values are any value larger than 8 and up to including 70,000,000 decimal (70 ms). The following larger values can be used as long as the output clock is synchronized to whole seconds as described in section "Synchronized Output Clock on SDP Pins": 125 ms; 250 ms and 500 ms.
Reserved	31:30	0x0	Reserved. Write 0x0, ignore on read.

8.15.20 Auxiliary Time Stamp 0 Register Low - AUXSTMPL0 (0xB65C; RW)

Field	Bit(s)	Initial Value	Description
TSTL	29:0	0x0	Auxiliary Time Stamp 0 LSB value (defined in ns units). RO field.
IO_TIMER_SEL	31:30	0x0	Select one of 4 1588 timers that the time stamp register is associated with. RW field.

8.15.21 Auxiliary Time Stamp 0 Register High -AUXSTMPH0 (0xB660; RO)

Reading this register releases the value stored in AUXSTMPH/L0 and enables timestamping of the next value.

Field	Bit(s)	Initial Value	Description
TSTH	31:0	0x0	Auxiliary Time Stamp 0 MSB value (defined in second units).

8.15.22 Auxiliary Time Stamp 1 Register Low AUXSTMPL1 (0xB664; RW)

Field	Bit(s)	Initial Value	Description
TSTL	29:0	0x0	Auxiliary Time Stamp 1 LSB value (defined in ns units). RO field.
IO_TIMER_SEL	31:30	0x0	Select one of 4 1588 timers that the time stamp register is associated with. RW field.

8.15.23 Auxiliary Time Stamp 1 Register High - AUXSTMPH1 (0xB668; RO)

Reading this register releases the value stored in AUXSTMPH/L1 and enables timestamping of the next value.

Field	Bit(s)	Initial Value	Description
TSTH	31:0	0x0	Auxiliary Time Stamp 1 MSB value (defined in second units).

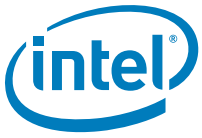
8.15.24 Time Sync RX Configuration - TSYNCRXCFG (0x5F50; RW)

Field	Bit(s)	Initial Value	Description
CTRLT	7:0	0x0	V1 control to timestamp.
MSGT	15:8	0x0	V2 Message Type to timestamp.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.15.25 Time Sync SDP Configuration Register - TSSDP (0x003C; RW)

This register defines the assignment of SDP pins to the time sync auxiliary capabilities.

Field	Bit(s)	Initial Value	Description
AUX0_SDP_SEL	1:0	00b	Select one of the SDPs to serve as the trigger for auxiliary time stamp 0 (AUXSTMPL0 and AUXSTMPH0 registers). 00b = SDP0 is assigned. 01b = SDP1 is assigned. 10b = SDP2 is assigned. 11b = SDP3 is assigned.
AUX0_TS_SDP_EN	2	0b	When set indicates that one of the SDPs can be used as an external trigger to Aux timestamp 0 (note that if this bit is set to one of the SDP pins, the corresponding pin should be configured to input mode using SPD_DIR).
AUX1_SDP_SEL	4:3	00b	Select one of the SDPs to serve as the trigger for auxiliary time stamp 1 (in AUXSTMPL1 and AUXSTMPH1 registers). 00b = SDP0 is assigned. 01b = SDP1 is assigned. 10b = SDP2 is assigned. 11b = SDP3 is assigned.
AUX1_TS_SDP_EN	5	0b	When set indicates that one of the SDPs can be used as an external trigger to Aux timestamp 1 (note that if this bit is set to one of the SDP pins, the corresponding pin should be configured to input mode using SPD_DIR).



Field	Bit(s)	Initial Value	Description
TS_SDP0_SEL	7:6	00b	SDP0 allocation to Tsync event – when TS_SDP0_EN is set, these bits select the Tsync event that is routed to SDP0. 00b = Target time 0 is output on SDP0. 01b = Target time 1 is output on SDP0. 10b = Freq clock 0 is output on SDP0. 11b = Freq clock 1 is output on SDP0.
TS_SDP0_EN	8	0b	When set indicates that SDP0 is assigned to Tsync.
TS_SDP1_SEL	10:9	00b	SDP1 allocation to Tsync event – when TS_SDP1_EN is set, these bits select the Tsync event that is routed to SDP1. 00b = Target time 0 is output on SDP1. 01b = Target time 1 is output on SDP1. 10b = Freq clock 0 is output on SDP1. 11b = Freq clock 1 is output on SDP1.
TS_SDP1_EN	11	0b	When set indicates that SDP1 is assigned to Tsync.
TS_SDP2_SEL	13:12	00b	SDP2 allocation to Tsync event – when TS_SDP2_EN is set, these bits select the Tsync event that is routed to SDP2. 00b = Target time 0 is output on SDP2. 01b = Target time 1 is output on SDP2. 10b = Freq clock 0 is output on SDP2. 11b = Freq clock 1 is output on SDP2.
TS_SDP2_EN	14	0b	When set indicates that SDP2 is assigned to Tsync.
TS_SDP3_SEL	16:15	00b	SDP3 allocation to Tsync event – when TS_SDP3_EN is set, these bits select the Tsync event that is routed to SDP3. 00b = Target time 0 is output on SDP3. 01b = Target time 1 is output on SDP3. 10b = Freq clock 0 is output on SDP3. 11b = Freq clock 1 is output on SDP3.
TS_SDP3_EN	17	0b	When set indicates that SDP3 is assigned to Tsync.
Reserved	31:18	0x0	Reserved. Write 0x0, ignore on read.

8.16 Time Sync Interrupt Registers

8.16.1 Time Sync Interrupt Cause Register - TSICR (0xB66C; RC/W1C)

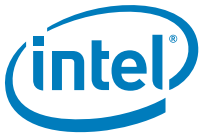
Note: Once *ICR.Time_Sync* is set, the internal value of this register should be cleared by writing 1b to all bits or cleared by a read to enable receiving an additional *ICR.Time_Sync* interrupt.

Field	Bit(s)	Initial Value	Description
SYS WARP 0	0	0b	SYSTIM0 Warp around. Set when SYSTIML0 wraparound This event should happen every second.
TXTS	1	0b	Transmit Timestamp. Set when new timestamp is loaded into *TXSTMP* register.
RXTS	2	0b	Receive Timestamp. Set when new timestamp is loaded into *RXSTMP* register.
TT0	3	0b	Target Time 0 Trigger. Set when target time 0 (*TRGTTIML/H0*) trigger occurs. This interrupt is enabled only if the EN_TT0 flag in the TSAUXC register is set. Note that this interrupt cause is set also by CLK0 output which is based on *TRGTTIM0.* When interrupt on CLK is needed the DIS_TS_CLEAR flag in the TSAUXC must be set to 1b.

Field	Bit(s)	Initial Value	Description
TT1	4	0b	Target Time 1 Trigger. Set when target time 1 (*TRGTTIML/H1*) trigger occurs. This interrupt is enabled only if the EN_TT1 flag in the TSAUXC register is set. Note that this interrupt cause is set also by CLK1 output which is based on *TRGTTIM1.* When interrupt on CLK is needed the DIS_TS_CLEAR flag in the TSAUXC must be set to 1b.
AUTT0	5	0b	Auxiliary Timestamp 0 Taken. Set when new timestamp is loaded into AUXSTMP 0 (auxiliary timestamp 0) register.
AUTT1	6	0b	Auxiliary Timestamp 1 Taken. Set when new timestamp is loaded into AUXSTMP 1 (auxiliary timestamp 1) register.
RSVD	7	0b	Reserved
RSVD	8	0b	Reserved
SYS WARP 1	9	0b	SYSTIML1 Warps around (expected every second).
SYS WARP 2	10	0b	SYSTIML2 Warps around (expected every second).
SYS WARP 3	11	0b	SYSTIML3 Warps around (expected every second).
RSVD	12	0b	Reserved
RSVD	13	0b	Reserved
RSVD	14	0b	Reserved
RSVD	15	0b	Reserved
RSVD	31:16	0x0	Reserved

8.16.2 Time Sync Interrupt Mask Register - TSIM (0xB674; RW)

Field	Bit(s)	Initial Value	Description
SYS WARP 0	0	0b	SYSTIM Warp Around 0 Mask. 0b = No interrupt generated for SYS WRAP 0 1b= Interrupt generated for SYS Wrap 0
TXTS	1	0b	Transmit Timestamp Mask. 0b = No interrupt generated for TXTS 1b= Interrupt generated for TXTS
RXTS	2	0b	Receive Timestamp Mask. 0b = No interrupt generated for RXTS 1b= Interrupt generated for RXTS
TT0	3	0b	Target time 0 Trigger Mask. 0b = No interrupt generated for TT0 1b= Interrupt generated for TT0
TT1	4	0b	Target time 1 Trigger Mask. 0b = No interrupt generated for TT1 1b= Interrupt generated for TT1
AUTT0	5	0b	Auxiliary Timestamp 0 Taken Mask. 0b = No interrupt generated for AUTT0 1b= Interrupt generated for AUTT0
AUTT1	6	0b	Auxiliary Timestamp 1 Taken Mask. 0b = No interrupt generated for AUTT1 1b = Interrupt generated for AUTT1



Field	Bit(s)	Initial Value	Description
RSVD	7	0b	Reserved
RSVD	8	0b	Reserved
SYS WARP 1	9	0b	SYSTIM Warp Around 1 Mask. 0b = No interrupt generated for SYS WARP 1 1b= Interrupt generated for SYS WARP 1
SYS WARP 2	10	0b	SYSTIM Warp Around 2 Mask. 0b = No interrupt generated for SYS WARP 2 1b= Interrupt generated for SYS WARP 2
SYS WARP 3	11	0b	SYSTIM Warp Around 3 Mask. 0b = No interrupt generated for SYS WARP 3 1b= Interrupt generated for SYS WARP 3
RSVD	12	0b	Reserved
RSVD	13	0b	Reserved
RSVD	14	0b	Reserved
RSVD	15	0b	Reserved
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

8.17 Time Sync - Preemption Statistics

8.17.1 Good TX Preempted Packets - PRMPTDTCNT (0x4280; RC)

Reset Source: Software Reset

Field	Bit(s)	Init.	Description
TCNT	31:0	0x0	Counts the total number of good transmitted preempted packets

8.17.2 TX Preemption event counter - PRMEVNTTCNT (0x4298; RC)

Reset Source: Software Reset

Field	Bit(s)	Init.	Description
TCNT	31:0	0x0	Counts the total transmitted number of preempted events

8.17.3 Good RX Preempted Packets - PRMPTDRCNT (0x4284; RC)

Reset Source: Software Reset

Field	Bit(s)	Init.	Description
RCNT	31:0	0x0	Counts the total number of good received preempted packets

8.17.4 RX Preemption event counter - PRMEVNTRCNT (0x429C; RC)

Reset Source: Software Reset

Field	Bit(s)	Init.	Description
RCNT	31:0	0x0	Counts the total number of good received preempted events

8.17.5 Good TX Preemptable Packets - PRMPBLTCNT (0x4288; RC)

Reset Source: Software Reset

Field	Bit(s)	Init.	Description
TCNT	31:0	0x0	Counts the total number of good transmitted non-preempted preemptable packets

8.17.6 Good RX Preemptable Packets - PRMPBLRCNT (0x428C; RC)

Reset Source: Software Reset

Field	Bit(s)	Init.	Description
RCNT	31:0	0x0	Counts the total number of good received non-preempted preemptable packets

8.17.7 Good TX Express Packets - PRMEXPTCNT (0x4290; RC)

Reset Source: Software Reset

Field	Bit(s)	Init.	Description
TCNT	31:0	0x0	Counts the total number of good transmitted excpress (non-preemptable) packets

8.17.8 Good RX Express Packets - PRMEXPRCNT (0x4294; RC)

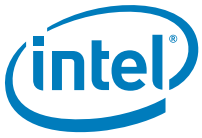
Reset Source: Software Reset

Field	Bit(s)	Init.	Description
RCNT	31:0	0x0	Counts the total number of good received Express (non-preemptable) packets

8.17.9 Preemption Exception Counter - PRMEXCPRCNT (0x42A0; RC)

Reset Source: Software Reset

Field	Bit(s)	Init.	Description
OOO_SMDC	7:0	0x0	Counts received packets with SMD-C AND NOT ReumeRx. The counter does not increments above 0xFF.



Field	Bit(s)	Init.	Description
OOO_FRAME_CNT	15:8	0x0	Counts received packets with SMD-C AND wrong Frame CNT. The counter does not increments above 0xFF.
OOO_FRAG_CNT	23:16	0x0	Counts received packets with SMD-C AND wrong Frag CNT. The counter does not increments above 0xFF.
MISS_FRAME_FRAG_CNT	31:24	0x0	Counts received packets with SMD-S AND ReumeRx. The counter does not increments above 0xFF.

8.18 Statistics Register Descriptions

All Statistics registers reset when read. In addition, they stick at 0xFFFF_FFFF when the maximum value is reached.

For the receive statistics it should be noted that a packet is indicated as received if it passes Foxville's filters and is placed into the packet buffer memory. A packet does not have to be transferred to host memory in order to be counted as received.

Due to divergent paths between interrupt-generation and logging of relevant statistics counts, it might be possible to generate an interrupt to the system for a noteworthy event prior to the associated statistics count actually being incremented. This is extremely unlikely due to expected delays associated with the system interrupt-collection and ISR delay, but might be observed as an interrupt for which statistics values do not quite make sense. Hardware guarantees that any event noteworthy of inclusion in a statistics count is reflected in the appropriate count within 1 μs; a small time-delay prior to a read of statistics might be necessary to avoid the potential for receiving an interrupt and observing an inconsistent statistics count as part of the ISR.

8.18.1 CRC Error Count - CRCERRS (0x4000; RC)

Field	Bit(s)	Initial Value	Description
CEC	31:0	0x0	CRC Error Count. Counts the number of receive packets with CRC errors. In order for a packet to be counted in this register, it must be 64 bytes or greater (from <Destination Address> through <CRC>, inclusively) in length. If receives are not enabled, then this register does not increment.

8.18.2 Alignment Error Count - ALGNERRC (0x4004; RC)

Field	Bit(s)	Initial Value	Description
AEC	31:0	0x0	Alignment Error Count. Counts the number of receive packets with alignment errors (the packet is not an integer number of bytes in length). In order for a packet to be counted in this register, it must be 64 bytes or greater (from <Destination Address> through <CRC>, inclusive) in length. If receives are not enabled, then this register does not increment. This register is valid only in MII mode during 10/100 Mb/s operation.

8.18.3 RX Error Count - RXERRC (0x400C; RC)

Field	Bit(s)	Initial Value	Description
RXEC	31:0	0x0	Rx Error Count Counts the number of packets received in which RX_ER was asserted by the PHY. In order for a packet to be counted in this register, it must be 64 bytes or greater (from <Destination Address> through <CRC>, inclusive) in length. If receives are not enabled, then this register does not increment.

8.18.4 Missed Packets Count - MPC (0x4010; RC)

Counts the number of missed packets. Packets are missed when the receive FIFO has insufficient space to store the incoming packet. This can be caused because of too few buffers allocated, or because there is insufficient bandwidth on the PCI bus. Events setting this counter causes *ICR.Rx Miss*, the Receiver Overrun interrupt, to be set. This register does not increment if receives are not enabled.

These packets are also counted in the Total Packets Received register as well as in Total Octets Received register.

Field	Bit(s)	Initial Value	Description
MPC	31:0	0x0	Missed Packets Count.

8.18.5 Single Collision Count - SCC (0x4014; RC)

This register counts the number of times that a successfully transmitted packet encountered a single collision. This register only increments if transmits are enabled (*TCTL.EN* is set) and Foxville is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
SCC	31:0	0x0	Number of times a transmit encountered a single collision.

8.18.6 Excessive Collisions Count - ECOL (0x4018; RC)

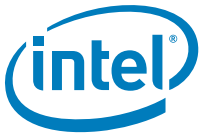
When 16 or more collisions have occurred on a packet, this register increments, regardless of the value of collision threshold. If collision threshold is set below 16, this counter won't increment. This register only increments if transmits are enabled (*TCTL.EN* is set) and Foxville is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
ECC	31:0	0x0	Number of packets with more than 16 collisions.

8.18.7 Multiple Collision Count - MCC (0x401C; RC)

This register counts the number of times that a transmit encountered more than one collision but less than 16. This register only increments if transmits are enabled (*TCTL.EN* is set) and Foxville is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
MCC	31:0	0x0	Number of times a successful transmit encountered multiple collisions.



8.18.8 Late Collisions Count - LATECOL (0x4020; RC)

Late collisions are collisions that occur after one slot time. This register only increments if transmits are enabled (*TCTL.EN* is set) and Foxville is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
LCC	31:0	0x0	Number of packets with late collisions.

8.18.9 Collision Count - COLC (0x4028; RC)

This register counts the total number of collisions seen by the transmitter. This register only increments if transmits are enabled (*TCTL.EN* is set) and Foxville is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
CCC	31:0	0x0	Total number of collisions experienced by the transmitter.

8.18.10 Receive Error Count - RERC (0x402C; RC)

Field	Bit(s)	Initial Value	Description
RERC	31:0	0x0	Total number of non-filtered packets received with errors. This includes: CRC error ; length error ; symbol error ; smd error (1); rx_data_error ; carrier extend error and rmii_alignment error (1) SMD errors includes the OOO_FRAME_CNT, OOO_FRG_CNT and MISS_FRAME_FRAG_CNT but exclude the OOO_SMDC.

8.18.11 Defer Count - DC (0x4030; RC)

This register counts defer events. A defer event occurs when the transmitter cannot immediately send a packet due to the medium being busy either because another device is transmitting, the IPG timer has not expired, half-duplex deferral events, reception of XOFF frames, or the link is not up. This register only increments if transmits are enabled (*TCTL.EN* is set). This counter does not increment for streaming transmits that are deferred due to TX IPG.

Field	Bit(s)	Initial Value	Description
CDC	31:0	0x0	Number of defer events.

8.18.12 Transmit with No CRS - TNCRS (0x4034; RC)

This register counts the number of successful packet transmissions in which the CRS input from the PHY was not asserted within one slot time of start of transmission from the MAC. Start of transmission is defined as the assertion of TX_EN to the PHY.

The PHY should assert CRS during every transmission. This register only increments if transmits are enabled (*TCTL.EN* is set). This register is not valid in SGMII mode, in full-duplex mode, and in 100 Mb/s half-duplex mode (HSD 359326).

Field	Bit(s)	Initial Value	Description
TNCRS	31:0	0x0	Number of transmissions without a CRS assertion from the PHY.

8.18.13 Host Transmit Discarded Packets by MAC Count - HTDPMC (0x403C; RC)

This register counts the number of packets sent by the host (and not the manageability engine) that are dropped by the MAC. This can include packets dropped because of excessive collisions or link fail events.

Field	Bit(s)	Initial Value	Description
HTDPMC	31:0	0x0	Number of packets sent by the host but discarded by the MAC.

8.18.14 Receive Length Error Count - RLEC (0x4040; RC)

This register counts receive length error events. A length error occurs if an incoming packet passes the filter criteria but is undersized or oversized. Packets less than 64 bytes are undersized. Packets over 1518, 1522 or 1526 bytes (according to the number of VLAN tags present) are oversized if Long Packet Enable (*RCTL.LPE*) is 0b. If *LPE* is 1b, then an incoming, packet is considered oversized if it exceeds the size defined in *RLPML.RLPML* field.

If receives are not enabled, this register does not increment. These lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusive. Packets sent to the manageability engine are included in this counter.

Note: Runt packets smaller than 25 bytes may not be counted by this counter.

Field	Bit(s)	Initial Value	Description
RLEC	31:0	0x0	Number of packets with receive length errors.

8.18.15 XON Received Count - XONRXC (0x4048; RC)

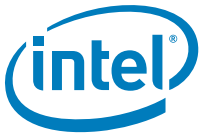
This register counts the number of valid XON packets received. XON packets can use the global address, or the station address. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
XONRXC	31:0	0x0	Number of XON packets received.

8.18.16 XON Transmitted Count - XONTXC (0x404C; RC)

This register counts the number of XON packets transmitted. These can be either due to a full queue or due to software initiated action (using TCTL.SWXOFF). This register only increments if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
XONTXC	31:0	0x0	Number of XON packets transmitted.



8.18.17 XOFF Received Count - XOFFRXC (0x4050; RC)

This register counts the number of valid XOFF packets received. XOFF packets can use the global address or the station address. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
XOFFRXC	31:0	0x0	Number of XOFF packets received.

8.18.18 XOFF Transmitted Count - XOFFTXC (0x4054; RC)

This register counts the number of XOFF packets transmitted. These can be either due to a full queue or due to software initiated action (using TCTL.SWXOFF). This register only increments if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
XOFFTXC	31:0	0x0	Number of XOFF packets transmitted.

8.18.19 FC Received Unsupported Count - FCRUC (0x4058; RC)

This register counts the number of unsupported flow control frames that are received.

The *FCRUC* counter increments when a flow control packet is received that matches either the reserved flow control multicast address (in the *FCAH/L* register) or the MAC station address, and has a matching flow control type field match (value in the *FCT* register), but has an incorrect op-code field. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Note: When the *RCTL.PMCF* bit is set to 1b then the *FCRUC* counter increments after receiving packets that don‚Äôt match standard address filtering.

Field	Bit(s)	Initial Value	Description
FCRUC	31:0	0x0	Number of unsupported flow control frames received.

8.18.20 Packets Received [64 Bytes] Count - PRC64 (0x405C; RC)

This register counts the number of good packets received that are exactly 64 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC64	31:0	0x0	Number of packets received that are 64 bytes in length.

8.18.21 Packets Received [65–127 Bytes] Count - PRC127 (0x4060; RC)

This register counts the number of good packets received that are 65-127 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC127	31:0	0x0	Number of packets received that are 65-127 bytes in length.

8.18.22 Packets Received [128–255 Bytes] Count - PRC255 (0x4064; RC)

This register counts the number of good packets received that are 128-255 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC255	31:0	0x0	Number of packets received that are 128-255 bytes in length.

8.18.23 Packets Received [256–511 Bytes] Count - PRC511 (0x4068; RC)

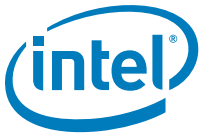
This register counts the number of good packets received that are 256-511 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC511	31:0	0x0	Number of packets received that are 256-511 bytes in length.

8.18.24 Packets Received [512–1023 Bytes] Count - PRC1023 (0x406C; RC)

This register counts the number of good packets received that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC1023	31:0	0x0	Number of packets received that are 512-1023 bytes in length.



8.18.25 Packets Received [1024 to Max Bytes] Count - PRC1522 (0x4070; RC)

This register counts the number of good packets received that are from 1024 bytes to the maximum (from <Destination Address> through <CRC>, inclusive) in length. The maximum is dependent on the current receiver configuration (for example, *RCTL.LPE*, etc.) and the type of packet being received. If a packet is counted in Receive Oversized Count, it is not counted in this register (refer to Section 8.18.37). This register does not include received flow control packets and only increments if the packet has passed address filtering and receives are enabled (*RCTL.RXEN* is set). Packets sent to the manageability engine are included in this counter.

Due to changes in the standard for maximum frame size for VLAN tagged frames in 802.3, Foxville accepts packets that have a maximum length of 1522 bytes. The RMON statistics associated with this range has been extended to count 1522 byte long packets. If *CTRL_EXT.EXT_VLAN* is set, packets up to 1526 bytes are counted by this counter.

Field	Bit(s)	Initial Value	Description
PRC1522	31:0	0x0	Number of packets received that are 1024-Max bytes in length.

8.18.26 Good Packets Received Count - GPRC (0x4074; RC)

This register counts the number of good packets received of any legal length. The legal length for the received packet is defined by the value of Long Packet Enable (*RCTL.LPE*) (refer to Section 8.18.37). This register does not include received flow control packets and only counts packets that pass filtering. This register only increments if receives are enabled (*RCTL.RXEN* is set). This register does not count packets counted by the Missed Packet Count (MPC) register. Packets sent to the manageability engine (MNGPRC) or dropped by the VMDq queueing process (SDPC) are included in this counter.

Note: *GPRC* can count packets interrupted by a link disconnect although they have a CRC error.

Field	Bit(s)	Initial Value	Description
GPRC	31:0	0x0	Number of good packets received (of any length).

8.18.27 Broadcast Packets Received Count - BPRC (0x4078; RC)

This register counts the number of good (no errors) broadcast packets received. This register does not count broadcast packets received when the broadcast address filter is disabled. This register only increments if receives are enabled (*RCTL.RXEN* is set). This register does not count packets counted by the Missed Packet Count (MPC) register. Packets sent to the manageability engine (MNGPRC) or dropped by the VMDq queueing process (SDPC) are included in this counter.

Field	Bit(s)	Initial Value	Description
BPRC	31:0	0x0	Number of broadcast packets received.

8.18.28 Multicast Packets Received Count - MPRC (0x407C; RC)

This register counts the number of good (no errors) multicast packets received. This register does not count multicast packets received that fail to pass address filtering nor does it count received flow control packets. This register only increments if receives are enabled (*RCTL.RXEN* is set). This register

does not count packets counted by the Missed Packet Count (MPC) register. Packets sent to the manageability engine (MNGPRC) or dropped by the VMDq queueing process (SDPC) are included in this counter.

Field	Bit(s)	Initial Value	Description
MPRC	31:0	0x0	Number of multicast packets received.

8.18.29 Good Packets Transmitted Count - GPTC (0x4080; RC)

This register counts the number of good (no errors) packets transmitted. A good transmit packet is considered one that is 64 or more bytes in length (from <Destination Address> through <CRC>, inclusively) in length. This does not include transmitted flow control packets. This register only increments if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
GPTC	31:0	0x0	Number of good packets transmitted.

8.18.30 Good Octets Received Count - GORCL (0x4088; RC)

These registers make up a 64-bit register that counts the number of good (no errors) octets received. This register includes bytes received in a packet from the <Destination Address> field through the <CRC> field, inclusive; *GORCL* must be read before *GORCH*.

In addition, it sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached. Only octets of packets that pass address filtering are counted in this register. This register does not count octets of packets counted by the Missed Packet Count (MPC) register. Octets of packets sent to the manageability engine are included in this counter. This register only increments if receives are enabled (*RCTL.RXEN* is set).

These octets do not include octets of received flow control packets.

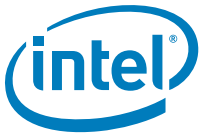
Field	Bit(s)	Initial Value	Description
GORCL	31:0	0x0	Number of good octets received ‚Äì lower 4 bytes.

8.18.31 Good Octets Received Count - GORCH (0x408C; RC)

Field	Bit(s)	Initial Value	Description
GORCH	31:0	0x0	Number of good octets received ‚Äì upper 4 bytes.

8.18.32 Good Octets Transmitted Count - GOTCL (0x4090; RC)

These registers make up a 64-bit register that counts the number of good (no errors) packets transmitted. This register must be accessed using two independent 32-bit accesses; GOTCL must be read before GOTCH.



In addition, it sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached. This register includes bytes transmitted in a packet from the <Destination Address> field through the <CRC> field, inclusive. This register counts octets in successfully transmitted packets that are 64 or more bytes in length. This register only increments if transmits are enabled (*TCTL.EN* is set).

These octets do not include octets in transmitted flow control packets.

Field	Bit(s)	Initial Value	Description
GOTCL	31:0	0x0	Number of good octets transmitted ,Äì lower 4 bytes.

8.18.33 Good Octets Transmitted Count - GOTCH (0x4094; RC)

Field	Bit(s)	Initial Value	Description
GOTCH	31:0	0x0	Number of good octets transmitted ,Äì upper 4 bytes.

8.18.34 Receive No Buffers Count - RNBC (0x40A0; RC)

This register counts the number of times that frames were received when there were no available buffers in host memory to store those frames (receive descriptor head and tail pointers were equal). The packet is still received if there is space in the FIFO. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Notes:

1. This register does not increment when flow control packets are received.
2. If a packet is replicated, this counter counts each of the packet that is dropped.

Field	Bit(s)	Initial Value	Description
RNBC	31:0	0x0	Number of receive no buffer conditions.

8.18.35 Receive Undersize Count - RUC (0x40A4; RC)

This register counts the number of received frames that passed address filtering, and were less than minimum size (64 bytes from <Destination Address> through <CRC>, inclusive), and had a valid CRC. This register only increments if receives are enabled (*RCTL.RXEN* is set). Packets sent to the manageability engine are included in this counter.

Note: Runt packets smaller than 25 bytes cannot be counted by this counter.

Field	Bit(s)	Initial Value	Description
RUC	31:0	0x0	Number of receive undersize errors.

8.18.36 Receive Fragment Count - RFC (0x40A8; RC)

This register counts the number of received frames that passed address filtering, and were less than minimum size (64 bytes from <Destination Address> through <CRC>, inclusive), but had a bad CRC (this is slightly different from the Receive Undersize Count register). This register only increments if receives are enabled (*RCTL.RXEN* is set). Packets sent to the manageability engine are included in this counter.

Note: Runt packets smaller than 25 bytes cannot be counted by this counter.

Field	Bit(s)	Initial Value	Description
RFC	31:0	0x0	Number of receive fragment errors.

8.18.37 Receive Oversize Count - ROC (0x40AC; RC)

This register counts the number of received frames with valid CRC field that passed address filtering, and were greater than maximum size. For definition of oversized packets, refer to Section 7.1.1.4.

If receives are not enabled, this register does not increment. These lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusive. Packets sent to the manageability engine are included in this counter.

Field	Bit(s)	Initial Value	Description
ROC	31:0	0x0	Number of receive oversize errors.

8.18.38 Receive Jabber Count - RJC (0x40B0; RC)

This register counts the number of received frames that passed address filtering, and were greater than maximum size and had a bad CRC (this is slightly different from the Receive Oversize Count register). For definition of oversized packets, refer to Section 7.1.1.4.

If receives are not enabled, this register does not increment. These lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusive. Packets sent to the manageability engine are included in this counter.

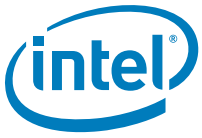
Field	Bit(s)	Initial Value	Description
RJC	31:0	0x0	Number of receive jabber errors.

8.18.39 Management Packets Received Count - MNGPRC (0x40B4; RC)

This register counts the total number of packets received that pass the management filters. Any packets with errors are not counted, except packets that are dropped because the management receive FIFO is full.

Packets sent to both the host and the management interface are not counted by this counter.

Field	Bit(s)	Initial Value	Description
MNGPRC	31:0	0x0	Number of management packets received.



8.18.40 Management Packets Dropped Count - MPDC (0x40B8; RC)

This register counts the total number of packets received that pass the management filters, that are dropped because the management receive FIFO is full. Management packets include any packet directed to the manageability console (for example, MC and ARP packets).

Field	Bit(s)	Initial Value	Description
MPDC	31:0	0x0	Number of management packets dropped.

8.18.41 Management Packets Transmitted Count - MNGPTC (0x40BC; RC)

This register counts the total number of transmitted packets originating from the manageability path.

Field	Bit(s)	Initial Value	Description
MPTC	31:0	0x0	Number of management packets transmitted.

8.18.42 BMC2OS Packets Sent by MC - B2OSPC (0x8FE0; RC)

This register counts the total number of transmitted packets sent from the manageability path that were sent to host. This includes packets received by the host and packet dropped in Foxville due to congestion conditions.

Counter is cleared when read by driver. Counter is also cleared by PCIe reset and Software reset. When reaching maximum value counter does not wrap-around.

Field	Bit(s)	Initial Value	Description
B2OSPC	31:0	0x0	BMC2OS packets sent by MC.

8.18.43 BMC2OS Packets Received by Host - B2OGPRC (0x4158; RC)

This register counts the total number of packets originating from the MC that reached the host.

If a packet is replicated, this counter counts each replication of the packet.

The counter clears when read by the software device driver. The counter also clears by a PCIe reset and software reset. When reaching the maximum, the value counter does not wrap-around.

Field	Bit(s)	Initial Value	Description
B2OGPRC	31:0	0x0	BMC2OS packets received by host.

8.18.44 OS2BMC Packets Received by MC - O2BGPTC (0x8FE4; RC)

This register counts the total number of packets originating from the host that reached the NC-SI interface.

The counter clears when read by the software device driver. The counter also clears by a PCIe reset and software reset. When reaching maximum value, the counter does not wrap-around.

Field	Bit(s)	Initial Value	Description
O2BGPTC	31:0	0x0	OS2BMC good packets transmitted count.

8.18.45 OS2BMC Packets Transmitted by Host - O2BSPC (0x415C; RC)

This register counts the total number of packets originating from the function that were sent to the manageability path. This includes packets received by the MC and packets dropped in Foxville due to congestion conditions.

Packets are dropped due to security reasons. For example, anti spoofing is not counted by this counter.

The counter is cleared when read by software device driver. The counter is also cleared by PCIe reset and software reset. When reaching a maximum value, the counter does not wrap-around.

Field	Bit(s)	Initial Value	Description
OS2BSPC	31:0	0x0	OS2BMC good packets transmit count.

8.18.46 Total Octets Received (Hi) - TORL (0x40C0; RC)

Field	Bit(s)	Initial Value	Description
TORL	31:0	0x0	Number of total octets received - Lower 4 bytes.

8.18.47 Total Octets Received - TORH (0x40C4; RC)

Field	Bit(s)	Initial Value	Description
TORH	31:0	0x0	Number of total octets received - Upper 4 bytes.

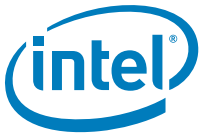
8.18.48 Total Octets Transmitted - TOTL (0x40C8; RC)

These registers make up a 64-bit register that counts the total number of octets transmitted. This register must be accessed using two independent 32-bit accesses; TOTL must be read before TOTH. This register sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached.

All transmitted packets have their octets summed into this register, regardless of their length or whether they are flow control packets. This register includes bytes transmitted in a packet from the <Destination Address> field through the <CRC> field, inclusive.

Octets transmitted as part of partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
TOTL	31:0	0x0	Number of total octets transmitted - lower 4 bytes.



8.18.49 Total Octets Transmitted (Hi) - TOTH (0x40CC; RC)

Field	Bit(s)	Initial Value	Description
TOTH	31:0	0x0	Number of total octets transmitted EnDash upper 4 bytes.

8.18.50 Total Packets Received - TPR (0x40D0; RC)

This register counts the total number of all packets received. All packets received are counted in this register, regardless of their length, whether they have errors, or whether they are flow control packets. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Notes:

1. Broadcast rejected packets are counted in this counter (as opposed to all other rejected packets that are not counted).
2. Runt packets smaller than 25 bytes cannot be counted by this counter.
3. *TPR* can count packets interrupted by a link disconnect although they have a CRC error.

Field	Bit(s)	Initial Value	Description
TPR	31:0	0x0	Number of all packets received.

8.18.51 Total Packets Transmitted - TPT (0x40D4; RC)

This register counts the total number of all packets transmitted. All packets transmitted are counted in this register, regardless of their length, or whether they are flow control packets.

Partial packet transmissions (collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard packets, packets received over the SMBus, and packets generated by the PT function.

Field	Bit(s)	Initial Value	Description
TPT	31:0	0x0	Number of all packets transmitted.

8.18.52 Packets Transmitted [64 Bytes] Count - PTC64 (0x40D8; RC)

This register counts the number of packets transmitted that are exactly 64 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (collisions in half-duplex mode) are not included in this register. This register does not include transmitted flow control packets (which are 64 bytes in length). This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard packets, packets received over the SMBus, and packets generated by the PT function.

Field	Bit(s)	Initial Value	Description
PTC64	31:0	0x0	Number of packets transmitted that are 64 bytes in length.

8.18.53 Packets Transmitted [65-127 Bytes] Count - PTC127 (0x40DC; RC)

This register counts the number of packets transmitted that are 65-127 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard packets, packets received over the SMBus, and packets generated by the PT function.

Field	Bit(s)	Initial Value	Description
PTC127	31:0	0x0	Number of packets transmitted that are 65-127 bytes in length.

8.18.54 Packets Transmitted [128-255 Bytes] Count - PTC255 (0x40E0; RC)

This register counts the number of packets transmitted that are 128-255 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard packets, packets received over the SMBus, and packets generated by the PT function.

Field	Bit(s)	Initial Value	Description
PTC255	31:0	0x0	Number of packets transmitted that are 128-255 bytes in length.

8.18.55 Packets Transmitted [256-511 Bytes] Count - PTC511 (0x40E4; RC)

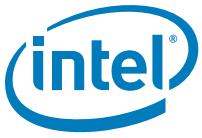
This register counts the number of packets transmitted that are 256-511 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets. Management packets must never be more than 200 bytes.

Field	Bit(s)	Initial Value	Description
PTC511	31:0	0x0	Number of packets transmitted that are 256-511 bytes in length.

8.18.56 Packets Transmitted [512-1023 Bytes] Count - PTC1023 (0x40E8; RC)

This register counts the number of packets transmitted that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets. Management packets must never be more than 200 bytes.

Field	Bit(s)	Initial Value	Description
PTC1023	31:0	0x0	Number of packets transmitted that are 512-1023 bytes in length.



8.18.57 Packets Transmitted [1024 Bytes or Greater] Count - PTC1522 (0x40EC; RC)

This register counts the number of packets transmitted that are 1024 or more bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set).

Due to changes in the standard for maximum frame size for VLAN tagged frames in 802.3, Springville transmits packets that have a maximum length of 1522 bytes. The RMON statistics associated with this range has been extended to count 1522 byte long packets. This register counts all packets. Management packets must never be more than 200 bytes. If CTRL.*EXT_VLAN* is set, packets up to 1526 bytes are counted by this counter.

Field	Bit(s)	Initial Value	Description
PTC1522	31:0	0x0	Number of packets transmitted that are 1024 or more bytes in length.

8.18.58 Multicast Packets Transmitted Count - MPTC (0x40F0; RC)

This register counts the number of multicast packets transmitted. This register does not include flow control packets and increments only if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
MPTC	31:0	0x0	Number of multicast packets transmitted.

8.18.59 Broadcast Packets Transmitted Count - BPTC (0x40F4; RC)

This register counts the number of broadcast packets transmitted. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets. Management packets must never be more than 200 bytes.

Field	Bit(s)	Initial Value	Description
BPTC	31:0	0x0	Number of broadcast packets transmitted count.

8.18.60 TCP Segmentation Context Transmitted Count - TSCTC (0x40F8; RC)

This register counts the number of TCP segmentation offload transmissions and increments once the last portion of the TCP segmentation context payload is segmented and loaded as a packet into the on-chip transmit buffer. Note that it is not a measurement of the number of packets sent out (covered by other registers). This register only increments if transmits and TCP segmentation offload are enabled.

This counter only counts pure TSO transmissions.

Field	Bit(s)	Initial Value	Description
TSCTC	31:0	0x0	Number of TCP Segmentation contexts transmitted count.

8.18.61 Interrupt Assertion Count - IAC (0x4100; RC)

This counter counts the total number of LAN interrupts generated in the system. In case of MSI-X systems, this counter reflects the total number of MSI-X messages that are emitted.

Field	Bit(s)	Initial Value	Description
IAC	31:0	0x0	This is a count of all the LAN interrupt assertions that have occurred.

8.18.62 Rx Packets to Host Count - RPTHC (0x4104; RC)

Field	Bit(s)	Initial Value	Description
RPTHC	31:0	0x0	This is a count of all the received packets sent to the host.

8.18.63 EEE TX LPI Count - TLPIC (0x4148; RC)

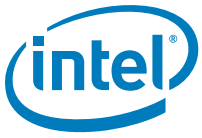
This register counts EEE TX LPI entry events. A EEE TX LPI event occurs when the transmitter enters EEE (IEEE802.3az) LPI state. This register only increments if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
ETLPIC	31:0	0x0	Number of EEE TX LPI events.

8.18.64 EEE RX LPI Count - RLPIC (0x414C; RC)

This register counts EEE RX LPI entry events. A EEE RX LPI event occurs when the receiver detects link partner entry into EEE (IEEE802.3az) LPI state. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
ERLPIC	31:0	0x0	Number of EEE RX LPI events.



8.18.65 Host Good Packets Transmitted Count-HGPTC (0x4118; RC)

Field	Bit(s)	Initial Value	Description
HGPTC	31:0	0x0	Number of good packets transmitted by the host.

This register counts the number of good (non-erred) packets transmitted sent by the host. A good transmit packet is considered one that is 64 or more bytes in length (from <Destination Address> through <CRC>, inclusively) in length. This does not include transmitted flow control packets or packets sent by the manageability engine. This register only increments if transmits are enabled (*TCTL.EN* is set).

8.18.66 Receive Descriptor Minimum Threshold Count-RXDMTC (0x4120; RC)

Field	Bit(s)	Initial Value	Description
RXDMTC	31:0	0x0	This is a count of the receive descriptor minimum threshold events.

This register counts the number of events where the number of descriptors in one of the Rx queues was lower than the threshold defined for this queue.

8.18.67 Host Good Octets Received Count - HGORCL (0x4128; RC)

Field	Bit(s)	Initial Value	Description
HGORCL	31:0	0x0	Number of good octets received by host ,Äì lower 4 bytes.

8.18.68 Host Good Octets Received Count - HGORCH (0x412C; RC)

Field	Bit(s)	Initial Value	Description
HGORCH	31:0	0x0	Number of good octets received by host ,Äì upper 4 bytes.

These registers make up a logical 64-bit register that counts the number of good (non-erred) octets received. This register includes bytes received in a packet from the <Destination Address> field through the <CRC> field, inclusive. This register must be accessed using two independent 32-bit accesses.; HGORCL must be read before HGORCH.

In addition, it sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached. Only packets that pass address filtering are counted in this register. This register counts only octets of packets that reached the host. The only exception is packets dropped by the DMA because of lack of descriptors in one of the queues. These packets are included in this counter.

This register only increments if receives are enabled (*RCTL.RXEN* is set).

8.18.69 Host Good Octets Transmitted Count (Lo) - HGOTCL (0x4130; RC)

Field	Bit(s)	Initial Value	Description
HGOTCL	31:0	0x0	Number of good octets transmitted by host - lower 4 bytes.

8.18.70 Host Good Octets Transmitted Count - HGOTCH (0x4134; RC)

Field	Bit(s)	Initial Value	Description
HGOTCH	31:0	0x0	Number of good octets transmitted by host - upper 4 bytes.

These registers make up a logical 64-bit register that counts the number of good (non-erred) packets transmitted. This register must be accessed using two independent 32-bit accesses. This register resets each time the upper 32 bits are read (HGOTCH).

In addition, it sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached. This register includes bytes transmitted in a packet from the <Destination Address> field through the <CRC> field, inclusive. This register counts octets in successfully transmitted packets which are 64 or more bytes in length. This register only increments if transmits are enabled (*TCTL.EN* is set).

These octets do not include octets in transmitted flow control packets or manageability packets.

8.18.71 Length Errors Count Register - LENERRS (0x4138; RC)

Field	Bit(s)	Initial Value	Description
LENERRS	31:0	0x0	Length error count.

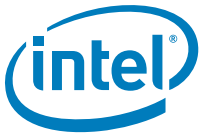
Note: Counts the number of receive packets with Length errors. For example, valid packets (no CRC error) with a *Length/Type* field with a value smaller or equal to 1500 greater than the frame size. In order for a packet to be counted in this register, it must pass address filtering and must be 64 bytes or greater (from <Destination Address> through <CRC>, inclusive) in length. If receives are not enabled, then this register does not increment.

8.18.72 Management Full Buffer Drop Packet Count - MNGFBDPC (0x4154; RC/W)

Field	Bit(s)	Initial Value	Description
MNGFDPC	31:0	0x0	Management Buffer Full Drop Packet Count. Counts the number of packets destined to management that were dropped due to lack of space in the management buffer. ***Note:*** The counter does not wrap around when reaching a value of 0xFFFFFFFF.

8.19 Per Queue Statistical Counters

Foxville supports nine statistical counters per queue.



8.19.1 Per Queue Good Packets Received Count - PQGPRC (0x10010 + n*0x100 [n=0...3]; RW)

This register counts the number of legal length good packets received in queue[n]. The legal length for the received packet is defined by the value of Long Packet Enable (*RCTL.LPE*) (refer to Section 8.18.37). This register does not include received flow control packets and only counts packets that pass filtering. This register only increments if receive is enabled.

Note: PQGPRC might count packets interrupted by a link disconnect although they have a CRC error.

Unlike some other statistics registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
GPRC	31:0	0x0	Number of good packets received (of any length).

8.19.2 Per Queue Good Packets Transmitted Count - PQGPTC (0x10014 + n*0x100 [n=0...3]; RW)

This register counts the number of good (no errors) packets transmitted on queue[n]. A good transmit packet is considered one that is 64 or more bytes in length (from <Destination Address> through <CRC>, inclusively) in length. This does not include transmitted flow control packets. This register only increments if transmits are enabled (*TCTL.EN* is set). This counter includes loopback packets or packets later dropped by the MAC.

A multicast packet dropped by some of the destinations, but sent to others is counted by this counter

Note: Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFFFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
GPTC	31:0	0x0	Number of good packets transmitted.

8.19.3 Per Queue Good Octets Received Count - PQGORC (0x10018 + n*0x100 [n=0...3]; RW)

This register counts the number of good (no errors) octets received on queue[n]. This register includes bytes received in a packet from the <Destination Address> field through the <CRC> field, inclusive.

Only octets of packets that pass address filtering are counted in this register. This register only increments if receive is enabled.

Note: CRC is part of the byte count if *DTXCTL.Count CRC* is set.

Note: Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
GORC	31:0	0x0	Number of good octets received.

8.19.4 Per Queue Good Octets Transmitted Count - PQGOTC (0x10034 + n*0x100 [n=0...3]; RW)

This register counts the number of good (no errors) packets transmitted on queue[n]. This register includes bytes transmitted in a packet from the <Destination Address> field through the <CRC> field, inclusive. Register also counts any padding that were added by the hardware. This register counts octets in successfully transmitted packets that are 64 or more bytes in length. Octets counted do not include octets in transmitted flow control packets. This register only increments if transmit is enabled.

A multicast packet dropped by some of the destinations, but sent to others is counted by this counter

Note: CRC is part of the byte count if *DTXCTL.Count CRC* is set.

Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
GOTC	31:0	0x0	Number of good octets transmitted ,Äì lower 4 bytes.

8.19.5 Per Queue Multicast Packets Received Count - PQMPRC (0x10038 + n*0x100 [n=0...3]; RW)

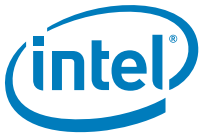
This register counts the number of good (no errors) multicast packets received on queue[n]. This register does not count multicast packets received that fail to pass address filtering nor does it count received flow control packets. This register only increments if receive is enabled.

Note: Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
MPRC	31:0	0x0	Number of multicast packets received.

8.19.6 Receive Queue Drop Packet Count - RQDPC (0xC030 + 0x40*n [n=0...3]; RW)

Field	Bit(s)	Initial Value	Description
RQDPC	31:0	0x0	Receive Queue Drop Packet Count. Counts the number of packets dropped by a queue due to lack of descriptors available. ***Note:*** Counter wraps around when reaching a value of 0xFFFFFFFF.



Packets dropped due to the queue being disabled might not be counted by this register.

8.19.7 Transmit Queue Drop Packet Count - TQDPC (0xE030 + 0x40*n [n=0...3]; RW)

Field	Bit(s)	Initial Value	Description
TQDPC	31:0	0x0	Transmit Queue Drop Packet Count. Counts the number of packets dropped by a queue due to lack of space in the loopback buffer or due to security (anti-spoof) issues. A multicast packet dropped by some of the destinations, but sent to others is counted by this counter. ***Note:*** Counter wraps around when reaching a value of 0xFFFFFFFF.

8.20 Wake Up Control Register Descriptions

8.20.1 Wake Up Control Register - WUC (0x5800; RW)

The *PME_En* and *PME_Status* bits of this register are reset when LAN_PWR_GOOD is 0b. When AUX_PWR = 0b, these register bits also reset by de-asserting PE_RST_N and during a D3 to D0 transition.

Note: If not mentioned otherwise inside the following table other bits are reset using the standard internal resets.

Field	Bit(s)	Initial Value	Description
APME	0	0b[1]	Advance Power Management Enable. If set to 1b, APM Wakeup is enabled. If this bit is set and the *APMPME* bit is cleared, reception of a magic packet asserts the *WUS.MAG* bit but does not assert a PME. ***Note:*** This bit is reset only on power-on reset (LAN_PWR_GOOD) but its value is auto-loaded from NVM on PCIe reset.
PME_En	1	0b	PME_En. This read/write bit is used by the software device driver to enable generation of a PME event without writing to the Power Management Control / Status Register (*PMCSR*) in the PCIe configuration space. ***Note:*** This bit reflects the value of the *PMCSR.PME_En* bit when the bit in the PMCSR register is modified. However, when the value of *WUC.PME_En* bit is modified by software device driver, the value is not reflected in the *PMCSR.PME_En* bit. ***Note:*** This bit is reset only on power-on reset (LAN_PWR_GOOD). When the AUX_PWR = 0b bit is also reset on de-assertion of PE_RST_N and during D3 to D0 transition.
PME_Status (RW1C)	2	0b	PME_Status. This bit is set when Foxville receives a wakeup event. It is the same as the PME_Status bit in the Power Management Control / Status Register (PMCSR). Writing a 1b to this bit clears also the *PME_Status* bit in the PMCSR. ***Note:*** This bit is reset only on power-on reset (LAN_PWR_GOOD). When the AUX_PWR = 0b bit is also reset on de-assertion of PE_RST_N and during D3 to D0 transition.

Field	Bit(s)	Initial Value	Description
APMPME	3	0b[1]	Assert PME On APM Wakeup. If set to 1b, Foxville sets the PME_Status bit in the Power Management Control / Status Register (PMCSR) and asserts PE_WAKE_N and sends a PM_PME PCIe message when APM Wakeup is enabled (*WUC.APME* = 1b) and Foxville receives a matching Magic Packet. ***Notes:*** 1. When *WUC.APMPME* is set PE_WAKE_N is asserted and a PM_PME message is sent even if *PMCSR.PME_En* is cleared. 2. This bit is reset only on power-on reset (LAN_PWR_GOOD) but its value is auto-loaded from NVM on SW reset.
PPROXYE	4	0b	Port Proxying Enable. When set to 1b Proxying of packets is enabled when device is in D3 low power state. ***Note:*** Proxy information and requirements is passed by the software device driver to firmware via the shared RAM host interface.
EN_APM_D0	5	0b[1]	Enable APM wake on D0. 0b = Enable APM wake only when function is in D3 and *WUC.APME* is set to 1b. 1b = Always enable APM wake when *WUC.APME* is set to 1b. ***Note:*** This bit is reset on power on reset (LAN_PWR_GOOD) only.
RoL_Mode	6	0b	RoL Mode 0b RoL is defined as a pulse of 50mSec 1b RoL is defined as a state and is only cleared by setting the SDP pin value by SW
SRST_PIN_EN	7	0b	Bit should be set to 1b to enable issuing a system reset on reception of a RoL packet via a SDP pin. The SDP pin chosen for this functionality needs to be defined in the WUC.SRST_PIN_SEL field.
SRST_PIN_SEL	9:8	00b	Field defines SDP pin used to issue System reset on reception of a RoL packet.
SRST_VAL	10	0b	SRST PIN Value. The value written to this bit specifies the non-active value of the SDP. SW needs to set the proper SDP value prior to RoL and post RoL to return to the "non active" value by setting the SRST_SET bit. Note that the SDP "non active" value should also load from the NVM as the SDP default value.
SRST_SET	11	0b	Set SRST value to SRST_VAL, this auto clear bit is used to set the SRST_PIN to the ‚Ä ƒ"non active" value by SW pre/post RoL event.
Reserved	31:12	0x0	Reserved. Write 0x0, ignore on read.

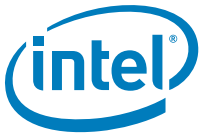
1. Loaded from the NVM.

8.20.2 Wakeup Filter Control Register - WUFC (0x5808; RW)

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.

If the *NoTCO* bit is set, then any packet that passes the manageability packet filtering, does not cause a wake-up event.

Field	Bit(s)	Initial Value	Description
LNKC	0	0b	Link Status Change Wakeup Enable.
MAG	1	0b	Magic Packet Wake-up Enable.
EX	2	0b	Directed Exact Wake-up Enable.[1]
MC	3	0b	Directed Multicast Wake-up Enable.
BC	4	0b	Broadcast Wake-up Enable.
ARP Directed	5	0b	ARP Request Packet and IP4AT Match Wake-up Enable. Wake on match of any ARP request packet that passed main filtering and Target IP address also matches one of the valid *IP4AT* filters.



Field	Bit(s)	Initial Value	Description
IPv4	6	0b	Directed IPv4 Packet Wake-up Enable.
IPv6	7	0b	Directed IPv6 Packet Wake-up Enable.
Reserved	8	0b	Reserved. Write 0b, ignore on read.
NS	9	0b	IPV6 Neighbor Solicitation Wake-up Enable. Wake on match of any NS packet that passed main filtering.
NS Directed	10	0b	IPV6 Neighbor Solicitation and Directed DA Match Wake-up Enable. Wake on match of NS packet and target IP address also matches *IPV6AT* filter.
ARP	11	0b	ARP Request Packet Wake-up Enable. Wake on match of any ARP request packet that passed main filtering.
Reserved	13:12	0x0	Reserved. Write 0x0, ignore on read.
FLEX_HQ	14	0b	Flex Filters Host Queuing 0b = Do not use flex filters for queueing decisions in D0 state. 1b = Use flex filters enabled in the WUFC register for queuing decisions in D0 state. ***Note:*** Should be enabled only when multi queueing is enabled (MRQC.Multiple Receive Queues = 010b or 000b).
NoTCO	15	0b	MPWU=criteriah=h.
FLX0	16	0b	Flexible Filter 0 Enable.
FLX1	17	0b	Flexible Filter 1 Enable.
FLX2	18	0b	Flexible Filter 2 Enable.
FLX3	19	0b	Flexible Filter 3 Enable.
FLX4	20	0b	Flexible Filter 4 Enable.
FLX5	21	0b	Flexible Filter 5 Enable.
FLX6	22	0b	Flexible Filter 6 Enable.
FLX7	23	0b	Flexible Filter 7 Enable.
FLX0_ACT	24	0b	Flexible Filter 0 Action. 0b= WoL. 1b= RoL
FLX1_ACT	25	0b	Flexible Filter 1 Action. 0b= WoL. 1b= RoL
FLX2_ACT	26	0b	Flexible Filter 2 Action. 0b= WoL. 1b= RoL
FLX3_ACT	27	0b	Flexible Filter 3 Action. 0b= WoL. 1b= RoL
Reserved	30:28	0b	Reserved.
FW_RST_WK	31	0b	Enable Wake on Firmware Reset Assertion. When set, a firmware reset causes a system wake so that the software driver can re-send proxying information to firmware.

1. If the *RCTL.UPE* is set, and the *EX* bit is also set, any unicast packet wakes up the system.

8.20.3 Wakeup Filter Control Register Extended - WUFC_EXT (0x0000580C; RW) MAC

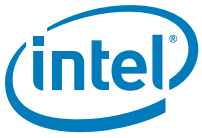
This register is used to enable each of the pre-defined and flexible filters for wakeup support. A value of 1b means the filter is turned on.; A value of 0b means the filter is turned off. If the NoTCO bit is set, then any packet that passes the manageability packet filtering as described in SectionHardSpace10.3, does not cause a Wake Up event.

Field	Bit(s)	Init.	Description
RSV1	7:0	0x0	
FLX8	8	0b	Flexible Filter 8 Enable.
FLX9	9	0b	Flexible Filter 9 Enable.
FLX10	10	0b	Flexible Filter 10 Enable.
FLX11	11	0b	Flexible Filter 11 Enable.
FLX12	12	0b	Flexible Filter 12 Enable.
FLX13	13	0b	Flexible Filter 13 Enable.
FLX14	14	0b	Flexible Filter 14 Enable.
FLX15	15	0b	Flexible Filter 15 Enable.
FLX16	16	0b	Flexible Filter 16 Enable.
FLX17	17	0b	Flexible Filter 17 Enable.
FLX18	18	0b	Flexible Filter 18 Enable.
FLX19	19	0b	Flexible Filter 19 Enable.
FLX20	20	0b	Flexible Filter 20 Enable.
FLX21	21	0b	Flexible Filter 21 Enable.
FLX22	22	0b	Flexible Filter 22 Enable.
FLX23	23	0b	Flexible Filter 23 Enable.
FLX24	24	0b	Flexible Filter 24 Enable.
FLX25	25	0b	Flexible Filter 25 Enable.
FLX26	26	0b	Flexible Filter 26 Enable.
FLX27	27	0b	Flexible Filter 27 Enable.
FLX28	28	0b	Flexible Filter 28 Enable.
FLX29	29	0b	Flexible Filter 29 Enable.
FLX30	30	0b	Flexible Filter 30 Enable.
FLX31	31	0b	Flexible Filter 31 Enable.

8.20.4 Wake Up Status Register - WUS (0x5810; RW1C)

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. 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 wakeup 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.

Field	Bit(s)	Initial Value	Description
LNKC	0	0b	Link Status Change.
MAG	1	0b	Magic Packet Received.
EX	2	0b	Directed Exact Packet Received. The packet's address matched one of the 32 pre-programmed exact values in the Receive Address registers (RAL[n]/RAH[n]), the packet was a unicast packet and *RCTL.UPE* is set to 1b.
MC	3	0b	Directed Multicast Packet Received. The packet was a multicast packet hashed to a value that corresponded to a 1 bit in the Multicast Table Array (MTA) or the packet was a multicast packet and *RCTL.MPE* is set to 1b.
BC	4	0b	Broadcast Packet Received.
ARP Directed	5	0b	ARP Request Packet with IPVA4AT filter Received. When set to 1b indicates a match on any ARP request packet that passed main filtering and Target IP address also matches one of the valid *IP4AT* filters.
IPv4	6	0b	Directed IPv4 Packet Received.
IPv6	7	0b	Directed IPv6 Packet Received.
MNG	8	0b	Indicates that a manageability event that should cause a PME happened.
NS	9	0b	IPV6 Neighbor Solicitation Received. When set to 1b indicates a match on any ICMPv6 packet such as Neighbor Solicitation (NS) packet or Multicast Listener Discovery (MLD) packet that passed main filtering.
NS Directed	10	0b	IPV6 Neighbor Solicitation with Directed DA Match Received. When set to 1b, indicates a match on any ICMPv6 packet such as a NS packet or MLD packet that passed main filtering and the field placed in the target IP address of a NS packet (9th byte to 24th byte of the ICMPv6 header) also matches a valid IPV6AT filter.
ARP	11	0b	ARP Request Packet Received. When set to 1b, indicates a match on an ARP request packet that passed main filtering.
Reserved	15:12	0x0	Reserved. Write 0bx0, ignore on read.
FLX0	16	0b	Flexible Filter 0 Match.
FLX1	17	0b	Flexible Filter 1 Match.
FLX2	18	0b	Flexible Filter 2 Match.
FLX3	19	0b	Flexible Filter 3 Match.
FLX4	20	0b	Flexible Filter 4 Match.
FLX5	21	0b	Flexible Filter 5 Match.
FLX6	22	0b	Flexible Filter 6 Match.
FLX7	23	0b	Flexible Filter 7 Match.
Reserved	30:24	0x0	Reserved. Write 0x0, ignore on read.
FW_RST_WK	31	0b	Wake Due to Firmware Reset Assertion Event. When set to 1b, indicates that a firmware reset assertion caused the system wake so that the software device driver can re-send proxying information to firmware.

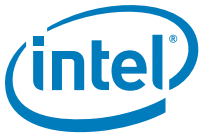
Note: FLX0-7 bits are set only when flex filter match is detected and WUFC.FLEX_HQ is 0b.

8.20.5 Wakeup Status Register Extended - WUS_EXT (0x5814; RW1/C)

This register is used to record statistics about all wakeup 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# is asserted. 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 wakeup filters, after the original wakeup packet is received, the WUS_EXT register is not updated with the new match detection until the register is cleared.

Field	Bit(s)	Initial value	Description
FLX0	0	0b	Flexible Filter 0 Match.
FLX1	1	0b	Flexible Filter 1 Match.
FLX2	2	0b	Flexible Filter 2 Match.
FLX3	3	0b	Flexible Filter 3 Match.
FLX4	4	0b	Flexible Filter 4 Match.
FLX5	5	0b	Flexible Filter 5 Match.
FLX6	6	0b	Flexible Filter 6 Match.
FLX7	7	0b	Flexible Filter 7 Match.
FLX8	8	0b	Flexible Filter 8 Match.
FLX9	9	0b	Flexible Filter 9 Match.
FLX10	10	0b	Flexible Filter 10 Match.
FLX11	11	0b	Flexible Filter 11 Match.
FLX12	12	0b	Flexible Filter 12 Match.
FLX13	13	0b	Flexible Filter 13 Match.
FLX14	14	0b	Flexible Filter 14 Match.
FLX15	15	0b	Flexible Filter 15 Match.
FLX16	16	0b	Flexible Filter 16 Match.
FLX17	17	0b	Flexible Filter 17 Match.
FLX18	18	0b	Flexible Filter 18 Match.
FLX19	19	0b	Flexible Filter 19 Match.
FLX20	20	0b	Flexible Filter 20 Match.
FLX21	21	0b	Flexible Filter 21 Match.
FLX22	22	0b	Flexible Filter 22 Match.
FLX23	23	0b	Flexible Filter 23 Match.
FLX24	24	0b	Flexible Filter 24 Match.
FLX25	25	0b	Flexible Filter 25 Match.
FLX26	26	0b	Flexible Filter 26 Match.
FLX27	27	0b	Flexible Filter 27 Match.
FLX28	28	0b	Flexible Filter 28 Match.
FLX29	29	0b	Flexible Filter 29 Match.
FLX30	30	0b	Flexible Filter 30 Match.
FLX31	31	0b	Flexible Filter 31 Match.



8.20.6 Wake Up Packet Length - WUPL (0x5900; RO)

This register indicates the length of the first wake-up packet received. It is valid if one of the bits in the Wakeup Status register (WUS) is set. It is not cleared by any reset.

Field	Bit(s)	Initial Value	Description
LEN	11:0	X	Length of Wake-up Packet. (If jumbo frames are enabled and the packet is longer than 2047 bytes then this field is 2047.)
Reserved	31:12	0x0	Reserved. Write 0x0, ignore on read.

8.20.7 Wake Up Packet Memory - WUPM (0x5A00 + 4*n [n=0...31]; RO)

This register is read-only and it is used to store the first 128 bytes of the wake up packet for software retrieval after system wake up. It is not cleared by any reset.

Field	Bit(s)	Initial Value	Description
WUPD	31:0	X	Wakeup Packet Data.

8.20.8 Wakeup Packet Memory Extended - WUPM_EXT[n] (0xB800 + 0x4*n, n=0...431; RO) MAC

This register is read-only and it is used to store the whole wakeup packet up to 1.5KByte for software retrieval after system wakeup. It is not cleared by any reset.
Memory size is 1728 Byte (432 x 32bit).

Field	Bit(s)	Initial Value	Description
WUPD	31:0	X	Wakeup Packet Data.

8.20.9 Proxying Filter Control Register - PROXYFC (0x5F60; RW)

This register is used to enable each of the pre-defined and flexible filters for proxying support. A value of 1b means the filter is turned on. A value of 0b means the filter is turned off.

If the *NoTCO* bit is set, then any packet that passes the manageability packet filtering, is not forwarded to management (i.e. firmware) for protocol offload even if it passes one of the proxying filters.

Field	Bit(s)	Initial Value	Description
D0_PROXY	0	0b	Enable Protocol Offload in D0. 0b = Enable protocol offload only when device is in D3 low power state. 1b = Enable protocol offload always. ***Note:*** Protocol offload is enabled only when the *WUC.PPROXYE* and *MANC.MPROXYE* bits are set to 1b.
Reserved	1	0b	Reserved. Write 0b, ignore on read.
EX	2	0b	Directed Exact Proxy Enable.[1]
MC	3	0b	Directed Multicast Proxy Enable.

Field	Bit(s)	Initial Value	Description
BC	4	0b	Broadcast Proxy Enable.
ARP Directed	5	0b	ARP Request Packet and IP4AT Match Proxy Enable. If set to 1b forward to Management for proxying on match of any ARP request packet that passed main filtering and Target IP address also matches one of the valid *IP4AT* filters.
IPv4	6	0b	Directed IPv4 Packet Proxy Enable.
IPv6	7	0b	Directed IPv6 Packet Proxy Enable.
Reserved	8	0b	Reserved. Write 0b, ignore on read.
NS	9	0b	IPV6 Neighbor Solicitation Proxy Enable. If set to 1b forward to management for proxying on match of any ICMPv6 packet such as a NS packet or MLD packet that passed main filtering.
NS Directed	10	0b	IPV6 Neighbor Solicitation and Directed DA Match Proxy Enable. If set to 1b forward to Management for proxying on match of any ICMPv6 packet such as a NS packet or MLD packet that passed main filtering and the field placed in the target IP address of a NS packet (9th byte to 24th byte of the ICMPv6 header) also matches a valid *IPV6AT* filter.
ARP	11	0b	ARP Request Packet Proxy Enable. If set to 1b, forwards to management for proxying on a match of any ARP request packet that passed main filtering.
Reserved	14:12	0x0	Reserved. Write 0x0, ignore on read.
NoTCO	15	0b	Ignore TCO/Management Packets for Proxying. 0b = Ignore only TCO/management packets for proxying that meet the criteria defined in the MNGONLY register (intended only for the MC and not the host). 1b = Ignore any TCO/management packets for proxying, even if in normal operation it's forwarded to the host in addition to the MC.
FLX0	16	0b	Flexible Filter 0 Enable.
FLX1	17	0b	Flexible Filter 1 Enable.
FLX2	18	0b	Flexible Filter 2 Enable.
FLX3	19	0b	Flexible Filter 3 Enable.
FLX4	20	0b	Flexible Filter 4 Enable.
FLX5	21	0b	Flexible Filter 5 Enable.
FLX6	22	0b	Flexible Filter 6 Enable.
FLX7	23	0b	Flexible Filter 7 Enable.
Reserved	31:24	0x0	Reserved. Write 0x0, ignore on read.

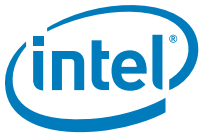
1. If the *RCTL.UPE* is set, and the *EX* bit is also set, any unicast packet is sent to management for proxying.

8.20.10 Proxying Status Register - PROXYS (0x5F64; RW1C)

This register is used to record statistics about all proxying 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. 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 matches one of the wake-up filters, after the original wake-up packet is received, the PROXYS register is updated with the matching filters accordingly.



Field	Bit(s)	Initial Value	Description
Reserved	1:0	0x0	Reserved. Write 0x0, ignore on read.
EX	2	0b	Directed Exact Packet Received. The packet's address matched one of the 32 pre-programmed exact values in the Receive Address registers, the packet was a unicast packet and *RCTL.UPE* is set to 1b.
MC	3	0b	Directed Multicast Packet Received. The packet was a multicast packet hashed to a value that corresponded to a 1 bit in the Multicast Table Array or the packet was a multicast packet and *RCTL.MPE* is set to 1b.
BC	4	0b	Broadcast Packet Received.
ARP Directed	5	0b	ARP Request Packet with IP4AT Filter Match Received. When set to 1b indicates a match on any ARP request packet that passed main filtering and Target IP address also matches one of the valid *IP4AT* filters.
IPv4	6	0b	Directed IPv4 Packet Received.
IPv6	7	0b	Directed IPv6 Packet Received.
Reserved	8	0b	Reserved. Write 0b, ignore on read.
NS	9	0b	IPV6 Neighbor Solicitation Received. When set to 1b, indicates a match on a NS packet that passed main filtering.
NS Directed	10	0b	IPV6 Neighbor Solicitation with Directed DA filter Match Received. When set to 1b, indicates a match on a NS packet and target IP address that also matches a valid *IPV6AT* filter.
ARP	11	0b	ARP Request Packet Received. When set to 1b indicates a match on any ARP request packet that passed main filtering.
Reserved	15:12	0x0	Reserved. Write 0x0, ignore on read.
FLX0	16	0b	Flexible Filter 0 Match.
FLX1	17	0b	Flexible Filter 1 Match.
FLX2	18	0b	Flexible Filter 2 Match.
FLX3	19	0b	Flexible Filter 3 Match.
FLX4	20	0b	Flexible Filter 4 Match.
FLX5	21	0b	Flexible Filter 5 Match.
FLX6	22	0b	Flexible Filter 6 Match.
FLX7	23	0b	Flexible Filter 7 Match.
Reserved	31:24	0b	Reserved. Write 0b, ignore on read.

Note: FLX0-7 bits are set only when flex filter match is detected and WUFC.FLEX_HQ is 0b.

8.20.11 Proxying Filter Control Extended Register - PROXYFCEX (0x5590; RW)

This register is an extension to PROXYFC and is used to control and enable the routing to management (i.e. firmware) of a set of pre-defined and flexible filters and filter combinations for proxying support.

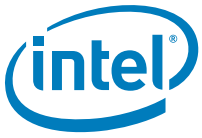
Field	Bit(s)	Initial Value	Description
mDNS	0	0b	Route to management if UDP and UDP port equals 5353.
mDNS_mDirected	1	0b	Route to management if UDP and port equals to 5353 and multicast IP match - if IPv4 224.0.0.251, if IPv6 FF02:FB.
mDNS_uDirected	2	0b	Route to management if UDP and port equals to 5353 and unicast IP match - if IPv4 any entry in IP4AT, if IPv6 any entry in IP6AT.
IPv4_mDirected	3	0b	Route to management if multicast IPv4 match 224.0.0.251.
IPv6_mDirected	4	0b	Route to management if multicast IPv6 match FF02::FB.
IGMP	5	0b	Route to management if IPv4 packet and protocol equals 02.
IGMP_mDirected	6	0b	Route to management if multicast IPv4 equals 224.0.0.251 and protocol equals 02.
ARP_RES	7	0b	ARP Response Packet Proxy Enable. If set to 1b forward to management for proxying on match of any ARP response packet that passed main filtering.
ARP_RES_Directed	8	0b	ARP Response Packet and IP4AT match Proxy Enable. If set to 1b forward to management for proxying on match of any ARP response packet that passed main filtering and target IP address also matches one of the valid *IP4AT* filters.
ICMPv4	9	0b	Route to management if IPv4 packet and protocol equals 01.
ICMPv4_Directed	10	0b	Route to management if unicast IPv4 equals any of the IP4AT addresses and the protocol equals 01.
ICMPv6	11	0b	Route to management if IPv6 packet and protocol equals 58.
ICMPv6_Directed	12	0b	Route to management if unicast IPv6 equals any of the IP6AT addresses and the protocol equals 58.
DNS	13	0b	Route to management if UDP/TCP and source port equals to 53.
Reserved	23:14	0x0	Reserved.
RA8	24	0b	Route to management if MAC address matched RA8.
RA9	25	0b	Route to management if MAC address matched RA9.
RA10	26	0b	Route to management if MAC address matched RA10.
RA11	27	0b	Route to management if MAC address matched RA11.
RA12	28	0b	Route to management if MAC address matched RA12.
RA13	29	0b	Route to management if MAC address matched RA13.
RA14	30	0b	Route to management if MAC address matched RA14.
RA15	31	0b	Route to management if MAC address matched RA15.

8.20.12 Proxying Extended Status Register - PROXYEXS (0x5594; RW1C)

This register is used to record statistics about all proxying 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. 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 matches one of the wake-up filters, after the original wake-up packet is received, the PROXYS register is updated with the matching filters accordingly.



Field	Bit(s)	Initial Value	Description
mDNS	0	0b	mDNS matched.
mDNS_mDirected	1	0b	mDNS_mDirected matched.
mDNS_uDirected	2	0b	mDNS_uDirected matched.
IPv4_mDirected	3	0b	IPv4_mDirected matched.
IPv6_mDirected	4	0b	IPv6_mDirected matched.
IGMP	5	0b	IGMP matched.
IGMP_mDirected	6	0b	IGMP_mDirected matched.
ARP_RES	7	0b	ARP_RES matched.
ARP_RES_Directed	8	0b	ARP_RES_Directed matched.
ICMPv4	9	0b	ICMPv4 matched.
ICMPv4_Directed	10	0b	ICMPv4_Directed matched.
ICMPv6	11	0b	ICMPv6 matched.
ICMPv6_Directed	12	0b	ICMPv6_Directed matched.
DNS	13	0b	DNS matched.
Reserved	23:14	0x0	Reserved. Write 0x0, ignore on read.
RA8	24	0b	RA8 matched.
RA9	25	0b	RA9 matched.
RA10	26	0b	RA10 matched.
RA11	27	0b	RA11 matched.
RA12	28	0b	RA12 matched.
RA13	29	0b	RA13 matched.
RA14	30	0b	RA14 matched.
RA15	31	0b	RA15 matched.

8.20.13 Wake Flex UDP/TCP Ports Filter - WFUTPF (0x5500 + 4*n [n=0...31]; RW)

Each 32-bit register (n=0...31) refers to one UDP/TCP port filters.

Field	Bit(s)	Initial Value	Description
Port	15:0	0x0	Flex TCP/UDP Destination Port Value.
Control	17:16	00b	Flex Port Control. 00b = Port filter disabled. 01b = UDP port. 10b =TCP port. 11b = TCP port and TCP flag SYN set, TCP flag RESET clear.
Action	18	0b	The *Action* bit defines the action to take on a match to an enabled filter. 0b= Host wake up. 1b= Route to the MC. Routing to the MC is only enabled when proxy functionality is enabled and is not intended for pass through. No host wake up performed.
Reserved	31:19	0x0	Reserved. Write 0x0, ignore on read.

8.20.14 Range Flex UDP/TCP Port Filter - RFUTPF (0x5580; RW)

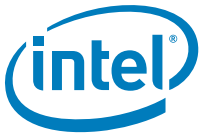
Field	Bit(s)	Initial Value	Description
LowPort	15:0	0x0	Range Flex UDP/TCP Ports Filter Low. This port filter marks the lowest port value for the range port filter.
HighPort	31:16	0x0	Range Flex UDP/TCP Ports Filter High. This port filter marks the highest port value for the range port filter.

8.20.15 Range and Wake Port Filter Control - RWPFC (0x5584; RW)

Field	Bit(s)	Initial Value	Description
RangeControl	1:0	00b	Range Port Filter Control. 00b = Port Filter disabled. 01b = UDP port. 10b = TCP port. 11b = TCP port and TCP flag SYN set; TCP flag RESET clear.
RangeAction	2	0b	The *Range Action* bit defines the action to take on a match to the range port filter. 0b = Host wake up. 1b = Route to the MC. Routing to the MC is only enabled when proxy functionality is enabled and is not intended for pass through.
Reserved	7:3	0x0	Reserved.
NonIPsecKA	8	0b	Non IPSEC Keep Alive to UDP 4500. Packet structure- UDP packet UDP destination port 4500, the first byte after the UDP header is not 0xFF. Refer to RFC 3948 for more information.
TCP_SSH_Data	9	0b	TCP SSH Data - port 22 (RESET, SYN, FIN - cleared). Packet structure- TCP packet with TCP destination port of 22; TCP flags doesn't have the RESET, SYN and FIN flag set.
MagicUDP	10	0b	UDP 3283 Magic WU Packet. Packet structure - DP packet UDP destination port 3283, first 2 bytes after the UDP header are 0x13, 0x88, UDP payload is >=100 bytes, and contains a Magic Packet structure in it.
Reserved	31:11	0x0	Reserved. Write 0x0, ignore on read.

8.20.16 Wake Flex UDP/TCP Ports Status - WFUTPS (0x5588, RW1C)

Field	Bit(s)	Initial Value	Description
Port0	0	0b	Flex Port 0 matched.
Port1	1	0b	Flex Port 1 matched.
Port2	2	0b	Flex Port 2 matched.
Port3	3	0b	Flex Port 3 matched.
Port4	4	0b	Flex Port 4 matched.
Port5	5	0b	Flex Port 5 matched.
Port6	6	0b	Flex Port 6 matched.



Field	Bit(s)	Initial Value	Description
Port7	7	0b	Flex Port 7 matched.
Port8	8	0b	Flex Port 8 matched.
Port9	9	0b	Flex Port 9 matched.
Port10	10	0b	Flex Port 10 matched.
Port11	11	0b	Flex Port 11 matched.
Port12	12	0b	Flex Port 12 matched.
Port13	13	0b	Flex Port 13 matched.
Port14	14	0b	Flex Port 14 matched.
Port15	15	0b	Flex Port 15 matched.
Port16	16	0b	Flex Port 16 matched.
Port17	17	0b	Flex Port 17 matched.
Port18	18	0b	Flex Port 18 matched.
Port19	19	0b	Flex Port 19 matched.
Port20	20	0b	Flex Port 20 matched.
Port21	21	0b	Flex Port 21 matched.
Port22	22	0b	Flex Port 22 matched.
Port23	23	0b	Flex Port 23 matched.
Port24	24	0b	Flex Port 24 matched.
Port25	25	0b	Flex Port 25 matched.
Port26	26	0b	Flex Port 26 matched.
Port27	27	0b	Flex Port 27 matched.
Port28	28	0b	Flex Port 28 matched.
Port29	29	0b	Flex Port 29 matched.
Port30	30	0b	Flex Port 30 matched.
Port31	31	0b	Flex Port 31 matched.

8.20.17 Wake Control Status - WCS (0x558C, RW1C)

Field	Bit(s)	Initial Value	Description
RangeControl	0	0b	RangeControl Matched.
NonIPsecKA	1	0b	NonIPsecKA Matched.
TCP_SSH_Data	2	0b	TCP_SSH_Data Matched.
MagicUDP	3	0b	MagicUDP Matched.
Reserved	29:4	0x0	Reserved. Write 0x0, ignore on read.
LocalIPOrNameConflict	30	0b	Local IP Conflict or Name Conflict Detected by Proxy. A firmware write of 1b sets the field, while a software write of 1b clears the field. Firmware writes to set are not blocked if other fields of the status are already set.
mDNS Proxy Error Recovery	31	0b	mDNS Proxy Error Recovery. A firmware write of 1b sets the field, while a software write of 1b clears the field. Firmware writes to set are not blocked if other fields of the status are already set.

8.20.18 IP Address Valid - IPAV (0x5838; RW)

The IP address valid indicates whether the IP addresses in the IP address table are valid.

Field	Bit(s)	Initial Value	Description
V40	0	0b	IPv4 Address 0 Valid.
V41	1	0b	IPv4 Address 1 Valid.
V42	2	0b	IPv4 Address 2 Valid.
V43	3	0b	IPv4 Address 3 Valid.
Reserved	15:4	0x0	Reserved. Write 0x0, ignore on read.
V60	16	0b	IPv6 Address 0 Valid.
Reserved	31:17	0x0	Reserved. Write 0x0, ignore on read.

8.20.19 IPv4 Address Table - IP4AT (0x5840 + 8*n [n=0...3]; RW)

The IPv4 address table is used to store the four IPv4 addresses for the ARP/IPv4 request packet and directed IP packet wake up.

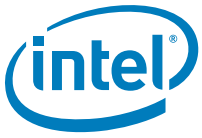
Field	Bit(s)	Initial Value	Description
IP Address	31:0	X	IPv4 Address n. ***Note:*** These registers are written in Big Endian order (LS byte is first on the wire and is the MS byte of the IPV4 address).

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV4ADDR0	0	0x5840	31:0	X	IPv4 Address 0.
IPV4ADDR1	2	0x5848	31:0	X	IPv4 Address 1.
IPV4ADDR2	4	0x5850	31:0	X	IPv4 Address 2.
IPV4ADDR3	6	0x5858	31:0	X	IPv4 Address 3.

8.20.20 IPv6 Address Table - IP6AT (0x5880 + 4*n [n=0...3]; RW)

The IPv6 address table is used to store the IPv6 addresses for neighbor discovery packet filtering and directed IP packet wake up.

Field	Bit(s)	Initial Value	Description
IP Address	31:0	X	IPv6 Address bytes 4*n+1:4*n +4. ***Note:*** These registers appear in Big Endian order (LS byte, LS address is first on the wire and is the MS byte of the IPV6 address).



Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV6ADDR0	0	0x5880	31:0	X	IPv6 Address 0, bytes 1-4.
	1	0x5884	31:0	X	IPv6 Address 0, bytes 5-8.
	2	0x5888	31:0	X	IPv6 Address 0, bytes 9-12.
	3	0x588C	31:0	X	IPv6 Address 0, bytes 16-13.

8.20.21 Flexible Host Filter Table Registers - FHFT (0x9000 + 256*n [n=0...3]; RW)

Each of the 32 Flexible Host Filters Table registers (FHFT and FHFT_EXT) contains 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 non-masked bytes in the FHFT register.

Each 128 byte filter is composed of 32 Data Dword entries, where each 2 Dwords are accompanied by an 8-bit mask, one bit per filter byte. When a bit in the 8-bit mask field is set the corresponding byte in the filter is compared.

The 8 LSB bits of the last Dword of each filter contains a length field defining the number of bytes from the beginning of the packet compared by this filter, the length field should be 8 bytes aligned value. If actual packet length is less than (length - 8) (length is the value specified by the length field), the filter fails. Otherwise, it depends on the result of actual byte comparison. The value should not be greater than 128.

Note: The length field must be 8 bytes aligned. For filtering packets shorter than 8 bytes aligned, the values should be rounded down to the previous 8 bytes aligned value.

Bits 31:8 of the last Dword of each filter also includes a *Queueing* field (refer to Section 8.20.21.2). When Foxville is in the D0 state, the *WUFC.FLEX_HQ* bit is set to 1b, *MRQC.Multiple Receive Queues* = 010b or 000b and the packet matches the flex filter, the *Queueing* field defines the receive queue for the packet, priority of the filter and actions to be initiated.

Table 8-20. FHFT Filter Description

31 8	7 0	31 8	7 0	31 0	31 0	Byte Offset
Reserved	Reserved	Reserved	Mask [7:0]	Data Dword 1	Data Dword 0	0x00
Reserved	Reserved	Reserved	Mask [15:8]	Data Dword 3	Data Dword 2	0x10
Reserved	Reserved	Reserved	Mask [23:16]	Data Dword 5	Data Dword 4	0x20
Reserved	Reserved	Reserved	Mask [31:24]	Data Dword 7	Data Dword 6	0x30
. . .						
Reserved	Reserved	Reserved	Mask [119:112]	Data Dword 29	Data Dword 28	0xE0
Queueing	Length	Reserved	Mask [127:120]	Data Dword 31	Data Dword 30	0xF0

Accessing the FHFT registers during filter operation can result in a packet being mis-classified if the write operation collides with packet reception. It is therefore advised that the flex filters are disabled prior to changing their setup.

8.20.21.1 Flex Filter - Example

Field	Address		Field	Address
Filter **0** DW 0	0x9000		Filter **0** DW 1	0x9004
Filter **0** Mask[7:0]	0x9008		Reserved	0x900C
Filter **0** DW 2	0x9010		Filter **0** DW 3	0x9004
. . .				
Filter **0** DW 30	0x90F0		Filter **0** DW 31	0x90F4
Filter **0** Mask[127:120]	0x90F8		Filter **0** Queuing ; Length	0x90FC
Filter **1** DW 0	0x9100		Filter **1** DW 1	0x9104
. . .				

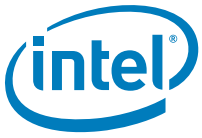
8.20.21.2 Flex Filter Queueing Field

The *Queueing* field resides in bits 31:8 of last Dword (Dword 63) of flex filter. The *Queueing* field defines the receive queue to forward the packet (*RQUEUE*), the filter priority (*FLEX_PRIO*) and additional filter actions. Operations defined in *Queueing* field are enabled when Foxville is in the D0 state, *MRQC.Multiple Receive Queues* = 010b or 000b, *WUFC.FLEX_HQ* is 1b and relevant *WUFC.FLX[n]* bit is set.

Field	Bit(s)	Initial Value	Description
Length	7:0	X	Length. Filter length in bytes. Should be 8 bytes aligned and not greater than 128 bytes.
RQUEUE	10:8	X	Receive Queue. Defines receive queue associated with this flex filter. When a match occurs in D0 state, the packet is forwarded to the receive queue.
Reserved	15:11	X	Reserved. Write 0x0, ignore on read.
FLEX_PRIO	18:16	X	Flex Filter Priority. Defines the priority of the filter assuming two filters with the same priority don't match. If two filters with the same priority match the incoming packet, the first filter (lowest address) is used in order to define the queue destination of this packet.
Reserved	23:19	X	Reserved. Write 0x0, ignore on read.
Immediate Interrupt	24	X	Enables issuing an immediate interrupt when the flex filter matches the incoming packet.
Drop	25	X	Pass / Drop option. 0b - packet that matches this filter pass the filter 1b - packet that matched this filter is dropped
Reserved	31:26	X	Reserved. Write 0x0, ignore on read.

8.20.22 Flexible Host Filter Table Extended Registers - FHFT_EXT (0x9A00 + 256*n [n=0...3]; RW)

Each of the four additional Flexible Host Filters table extended registers (FHFT_EXT) contains a 128 byte pattern and a corresponding 128-bit mask array. The structure and functionality of the FHFT_EXT table is identical to the FHFT table.



8.20.23 Flex Filter indirect table select - FHFTSL (0x5804; RW) MAC

Indirect control for FHFT - wakeup flex filters.

This register is programmed by the "Set Filter Indirect Table Select" host slave command described in Section 10.6.3.5. This register is required only when the software need to access more than 8 flexible filters. If less equal than 8 filters are required, the software can access them through the FHFT and FHFT_EXT registers as is.

Note: In step C the host has direct access to FHFTSL and the use of host slave command is not required.

Field	Bit(s)	Initial Value	Description
FTSL	1:0	0x0	Indirect address Select for 8 filter table out of 32. (FHFT, FHFT_EXT Address) b00: Select filters 0-7 (default) b01: Select filters 15-8 b10: Select filters 23-16 b11: Select filters 31-24
RESERVED	31:2	0x0	

8.21 Management Register Descriptions

All management registers are controlled by the remote MC for both read and write. Host accesses to the management registers are blocked for write unless debug write is enabled via straping options. The attributes for the fields in this section refer to the MC access rights.

Note: All the registers described in this section can get their default values from the NVM when manageability pass through works in legacy SMBus mode. The only exception being the MANC register where part of the bits are masked. The specific MANC bits that can be loaded from the NVM are indicated in the register description.

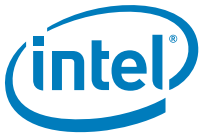
8.21.1 Management Control Register - MANC (0x5820; RW)

The MANC register can be written by the MC and is not accessible to the host for writing.

Reset by LAN_PWR_GOOD

Field	Bit(s)	Initial Value	Description
Flow Control Discard	0	0b	0b = Apply filtering rules to packets with Flow Control EtherType. 1b = Discard packets with Flow Control EtherType. ***Note:*** Flow Control EtherType is 0x8808.
NCSI Discard	1	0b	0b = Apply filtering rules to packets with NC-SI EtherType. 1b = Discard packets with NC-SI EtherType. ***Note:*** NC-SI EtherType is 0x88F8.
Reserved	13:2	0x0	Reserved. Write 0x0, ignore on read.

Field	Bit(s)	Initial Value	Description
FW_RESET (RW1C)	14	0b	Firmware Reset Occurred. Set to 1b on a TCO firmware reset. ***Notes:*** Cleared by write 1b. 1. BMC can read this bit using the Read Configuration OEM command. Bit is reset by LAN_PWR_GOOD only.
TCO_Isolate (RO)	15	0b	Set to 1b on a TCO Isolate command. When the *TCO_Isolate* bit is set. Host write cycles are completed successfully on the PCIe but silently ignored by internal logic. Note that when firmware initiates the TCO Isolate command it also initiates a firmware interrupt via the *ICR.MNG* bit to the host and writes a value of 0x22 to the*FWSM.Ext_Err_Ind* field. ***Notes:*** This bit is RO and mirrors the value of the *Isolate* bit in the internal management registers. 1. Bit is reset by LAN_PWR_GOOD and FW reset only. Bit reflects internal management Aux register bit.
TCO_RESET (RW1C)	16	0b	TCO Reset Occurred. Set to 1b on a TCO reset, to reset LAN port by the MC. ***Notes:*** Cleared by write 1b. 1. BMC can read this bit using the Read Configuration OEM command. Bit is reset by LAN_PWR_GOOD only.
RCV_TCO_EN	17	0b[1]	TCO Receive Traffic Enabled. When this bit is set, receive traffic to manageability is enabled. This bit should be set only if either the MANC.EN_BMC2OS or MANC.EN_BMC2NET bits are set.
KEEP_PHY_LINK_UP (VETO)	18	0b[1]	Block PHY reset and power state changes. When this bit is set, the PHY reset and power state changes do not effect the PHY, This bit can not be written to unless the *Keep_PHY_Link_Up_En* NVM bit is set.Bit is only reset by LAN_POWER_GOOD.
RCV_ALL	19	0b[1]	Receive All Enable (promiscuous mode) When set, all the traffic received from the wire is forwarded to the manageability. This bit should be set only for debug purposes.
Inhibit ULP	20	0b	Inhibit ULP when set to '1'.
Reserved	22:21	0b	Reserved. Write 0b, ignore on read.
EN_XSUM_FILTER	23	0b[1]	Enable Checksum Filtering to MNG. When this bit is set, only packets that pass L3and L4 checksums are sent to the manageability block.
EN_IPv4_FILTER	24	0b[1]	Enable IPv4 address Filters.
FIXED_NET_TYPE	25	0b[1]	Fixed Net 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 can be forwarded to the manageability engine.
NET_TYPE	26	0b[1]	Net Type. 0b = Pass only un-tagged packets. 1b = Pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE is set.
IPV6_ADV_ONLY	27	0b	Reserved. When set, any Transmit IPV6 packet using legacy descriptor is dropped.



Field	Bit(s)	Initial Value	Description
EN_BMC2OS (RO)	28	0b[1]	Enable MC-to-OS and OS-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 the host is the MAC. The address filter and VLANs (RAH/L, MTA, VFTA and VLVF registers) indicate that it should. When cleared, the operating system traffic is not forwarded to the MC even if the decision filters indicates 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.
EN_BMC2NET (RO)	29	1b[1]	Enable 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, the 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. ***Note:*** 1. Initial value loaded according to value of *Port n traffic types* field in NVM. 2. Bit reflects internal management Aux register bit. 3. This bit can change while the host is sending or receiving traffic.
MPROXYE (RO)	30	0b[1]	Management Proxying Enable. When set to 1b, proxying of packets is enabled when the device is in a D3 low power state. 0b = Manageability does not support proxying. 1b = Manageability supports proxying. Note: Bit reflects internal Management Aux register bit. ***Note:*** Proxy information and requirements are passed by the software device driver to firmware via the shared RAM host interface. ***Note:*** Proxying traffic from and to firmware is not affected by the *MANC.RCV_TCO_EN* bit or the MANC.EN_BMC2NET bit.
MPROXYA (RO)	31	0b	Management Proxying Active (This flag reflects an internal MNG Aux register). 0b = Proxy is not active for the Manageability 1b = Proxy is active for the Manageability

1. Bit loaded from NVM by Firmware.

8.21.2 Management Only Traffic Register - MNGONLY (0x5864; RW)

The MNGONLY register enables exclusive filtering of a certain type 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 be forwarded to the host even if they pass the host L2 filtering process.

Each manageability decision filter has a corresponding bit in the MNGONLY register. When a manageability decision filter forwards a packet to manageability, it can also block the packet from being forwarded to the host if the corresponding *MNGONLY* bit is set.

Reset by LAN_PWR_GOOD

Field	Bit(s)	Initial Value[1]	Description
Exclusive to MNG	7:0	0x0	Exclusive to MNG. When set, indicates that packets forwarded by the manageability filters to manageability are not sent to the host.
Reserved	31:8	0x0	Reserved. Write 0x0, ignore on read.

1. The initial values for this register can be loaded from the NVM by Firmware after power-up reset or firmware reset.

8.22 Host Slave Interface Registers Description

The software device driver communicates with the manageability block through CSR access. The manageability block is mapped to address 0x8800 -0x8FFF on the slave bus.

8.22.1 Host Slave Command Interface to Manageability Firmware

This interface is used by the software device driver for several of commands and for delivering various types of data structure in both directions (MNG →Host, Host → MNG).

The address space is separated into two areas:

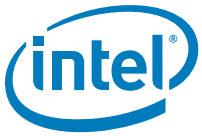
1. Direct access to the internal Management (ARC)DATA RAM: The internal DATA RAM is mapped to address 0x8800-0x8EFF. Writing to this address space goes directly to the RAM.
2. Control registers located at address 0x8F00.

8.22.1.1 Host Slave Command I/F Flow

This interface is used for the external host software to access the MMS sub-system. The host software can write a command block or read data structure directly from the DATA RAM. The host software controls these transactions through a slave access to the control register.

The following flow describes the process of initiating a command to the MMS:

1. The software device driver takes ownership of the *SW_FW_SYNC.SW_MNG_SM* bit according to the flow described in Section 4.8.1.
2. The software device driver reads the HICR register and checks that the enable bit is set.
3. The software device driver writes the relevant command block into the shared RAM area.
4. The software device driver sets the *Command* bit in the control register. Setting this bit causes an interrupt to management (can be masked).
5. The software device driver polls the Control register until the *Command* bit is cleared by hardware.
6. When the MMS is done with the command, it clears the *Command* bit (if the MMS should reply with a data, it should clear the bit only after the data is in the RAM area where the software device driver can read it).
7. If the software device driver reads the Control register and the *SV* bit is set, it means that there is a valid status of the last command in the RAM. If the *SV* is not set it means that the command has failed with no status in the RAM.



8.22.2 HOST Interface Control Register - HICR (0x8F00; RW)

Register reset by LAN_PWR_GOOD only.

Field	Bit(s)	Initial Value	Description
En (RO)	0	0b	Enable. When set, it indicates that a RAM area is provided for software device driver accesses. This bit is read only for the software device driver.
C	1	0b	Command. The software device driver sets this bit when it has finished putting a command block in the management (ARC) internal DATA RAM. This bit should be cleared by the firmware after the command's processing completes. Setting this bit causes an interrupt to the ARC.
SV (RO)	2	0b	Status Valid. Indicates that there is a valid status in CSR area that the software device driver can read. 1b = status valid. 0b = status not valid. The value of the bit is valid only when the *C* bit is cleared. Only the software device driver reads this bit.
Reserved	3	0b	Reserved.
SWSMB	4	0b	Software/firmware semaphore - Software bit. This bit is set only by the HOST software (Read only to the firmware). The bit is not set if bit 8 in the software status register (CSR 0x8F10) is set. This bit can be used by the software and firmware to synchronize mutual resources. ***Note:*** This bit is cleared by hardware on PCIe reset or Device Reset (*CTRL.DEV_RST*) reset or manageability watchdog (WD) expire scenario.
Reserved	5	0b	Reserved Write 0, ignore on read.
Reserved	~~6~~7:4	0x0	Reserved.
Mailbox Mode Enable (RO)	8	0b	Mailbox Mode Enable. When set it indicates that Mailbox register (*FWSWMB*) is provided for device driver.
Memory Base Enable (RO)	9	0b	Enable host access to memory base register. This bit is set by the firmware and is read only to the software device driver.
Reserved	31:10	0x0	Reserved. Write 0x9, ignore on read.

8.22.3 Firmware Software Mailbox Register- FWSWMB (0x8F04; RW)

Register reset by LAN_PWR_GOOD only.

Field	Bit(s)	Initial Value	Description
Mailbox	31:0	0x0	Mailbox: scratch pad that can be used for software/firmware communication. Has no impact on Foxville hardware behavior.

8.22.4 Firmware Status Register - FWSTS (0x8F0C; RW)

This register is connected directly to the Management (ARC aux) register space. The read and write to this register by the firmware is done by AUX registers access commands.

Register reset by LAN_PWR_GOOD only.

Field	Bit(s)	Initial Value	Description
FWSW (RO)	15:0	0x0	Firmware Status word This bit is read only through the HOST interface.
Reserved	30:16	0x0	Reserved Write 0, ignore on read.
FWRI (RW1C)	31	1b	Firmware Reset Indication. Set when firmware reset is asserted. Cleared when HOST writes 1b to it. Writing 0b to this bit does not change its value. Note: This bit is also set after LAN_POWER_GOOD.

8.22.5 Software Status Register - SWSR (0x8F10; RW)

This register is connected directly to the Management (ARC aux) register space. The read and write to this register by the firmware is done by AUX registers access commands.

Register reset by LAN_PWR_GOOD, PCIe Reset only or Device Reset (*CTRL.DEV_RST*).

Field	Bit(s)	Initial Value	Description
SWS	7:0	0x0	Software Status. These bits can be written only by the HOST.
FWSMB (RO)	8	0b	Software/firmware Semaphore - Firmware bit. This bit is set only by the firmware (Read only to the HOST software). The bit is not set if bit 4 in the HOST Interface Control Register (CSR 0x8F00) is set. The firmware should set this bit and then read it to see if it was set. If it was set, it means that the firmware owns the semaphore. This bit can be used to synchronize software/firmware mutual resources. This bit can be written only by the firmware.
Reserved	31:9	0x0	Reserved. Write 0, ignore on read.

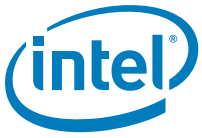
8.22.6 Message to Manageability Register - MSG2MNG (0x8F50; RCW) MNG

Register reset by LAN_PWR_GOOD, PCIe Reset and CTRL.DEV_RESET

Setting any unmasked bit in the MSG2MNG register causes the "Host Command" interrupt to the MNG engine.

The read and write to this register by the firmware is done by AUX registers access commands.

Field	Bit(s)	Initial Value	Description
SWMSG	23:0	0x0	SW message bits (write '1' to set). Bits are cleared by the device after handling the request.



PTMCOMP	24	0b	PTM Cycle Completion. This flag is set by the hardware and it is cleared by the firmware after handling the request.
OTHERCAUSE	25	0b	Other HW Cause. This flag is set by the hardware and it is cleared by the firmware after handling the request.
RESERVED	31:26	0x0	Reserved

8.22.7 Message to Manageability Mask Register - MSKMSG2MNG (0x8F54; RW) MNG

Register reset by LAN_PWR_GOOD, PCIe Reset and CTRL.DEV_RESET

Field	Bit(s)	Initial Value	Description
SWMSG	23:0	0x0	SW message Mask bits.
PTMCOMP	24	0b	PTM Cycle Completion Mask bit
OTHERCAUSE	25	0b	Other HW Cause Mask bit
RESERVED	31:26	0x0	Reserved

8.22.8 Host Interface Buffer Base Address - HIBBA (0x8F40; RW)

This register is reset by a firmware reset.

Notes:

8. This resister is accessible to the host driver only if *Memory Base Enable* is set in HICR; otherwise, the register is read only to the host driver.

Field	Bit(s)	Initial Value	Description
BA	19:0	0x17800	Host interface buffer base address in the device internal memory space (in bytes). Base address for the CSR slave access. The address must be 1 KB aligned (bits 9:0 are RO hardwired to zero).
Reserved	31:20	0x0	Reserved. Write 0x0, ignore on read.

8.22.9 Host Interface Buffer Maximum Offset - HIBMAXOFF (0x8F44; RO)

The register holds the maximum offset in bytes in the memory buffer that the host can access from address 0x8800 in its address space. Any access above this value is blocked by hardware.

This register is reset by a firmware reset.

Field	Bit(s)	Initial Value	Description
MAXOFF	9:0	0x3FF	Maximum offset in the HIB for the CSR slave access. The 2 LSBs are always set to 11b.
Reserved	31:10	0x0	Reserved. Write 0x0, ignore on read.

8.23 Memory Error Registers Description

Main internal memories are protected by Error Correcting Code (ECC) or parity bits. Foxville contains several registers that enable and report detection of internal memory errors. Description and usage of these registers can be found in Section 7.7.

8.23.1 Parity and ECC Error Indication - PEIND (0x1084; RC)

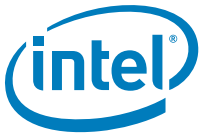
Register reset by LAN_PWR_GOOD only.

Field	Bit(s)	Initial Value	Description
lanport_parity_fatal_ind (LH)	0	0b	Fatal Error detected in LAN port memory. Bit is latched high and cleared on read.
mng_parity_fatal_ind (RC)	1	0b	Fatal Error detected in management memory. Bit is latched high and cleared on read.
pcie_parity_fatal_ind (RC)	2	0b	Fatal Error detected in PCIe memory. Bit is latched high and cleared on read.
dma_parity_fatal_ind (RC)	3	0b	Fatal Error detected in DMA memory. Bit is latched high and cleared on read.
Reserved	31:4	0x0	Reserved. Write 0x0 ignore on read.

8.23.2 Parity and ECC Indication Mask - PEINDM (0x1088; RW)

Register reset by LAN_PWR_GOOD only.

Field	Bit(s)	Initial Value	Description
LANPORT_PARITY_FATAL_IND	0	1b	When set and PEIND.lanport_parity_fatal_ind is set, enable interrupt generation by setting the ICR.FER bit.
MANG_PARITY_FATAL_IND	1	1b	When set and PEIND.mng_parity_fatal_ind is set, enable interrupt generation by setting the ICR.FER bit.
PCIE_PARITY_FATAL_IND	2	1b	When set and PEIND.pcie_parity_fatal_ind is set, enable interrupt generation by setting the ICR.FER bit.
DMA_PARITY_FATAL_IND	3	1b	When set and PEIND.dma_parity_fatal_ind is set, enable interrupt generation by setting the ICR.FER bit.
RESERVED	31:4	0x0	Reserved. Write 0x0, ignore on read.



8.23.3 Packet Buffer ECC Status - PBECCSTS (0x245c; RW)

Field	Bit(s)	Init.	Description
ecc_en	0	0x1	ECC Enable.
ins_pb_ecc_err(SC)	1	0x0	Insert ECC error to the DBU RAM.
pb_cor_err_sta(RW1C)	2	0x0	DBU RAM correctable error indication. Bit is clean by write 1b.
Reserved	31:3	0x0	Reserved. Write 0x0, ignore on read.

8.23.4 PCIe Parity Control Register - PCIEERRCTL (0x5BA0; RW)

Reset by LAN_PWR_GOOD and PCIe reset.

Field	Bit(s)	Initial Value	Description
GPAR_EN	0	0b[1]	Global Parity Enable. When cleared, parity checking of all RAMs is disabled. ***Note:*** This bit resets only at LAN_PWR_GOOD.
Reserved	5:1	01000b	Reserved. Write 0x0, ignore on read.
ERR EN RX CDQ 0	6	1b	RX CDQ 0 Parity Check Enable If set to 1b enable Inbound on Completion Data 0 ram parity check.
ERR INJ RX CDQ 0 (SC)	7	0b	RX CDQ 0 ERR Injection Inject parity error to Inbound on Completion Data 0 ram.
ERR EN RX CDQ 1	8	1b	RX CDQ 1 Parity Check Enable. If set to 1b enable Inbound on Completion Data 1 ram parity check.
ERR INJ RX CDQ 1 (SC)	9	0b	RX CDQ 1 ERR Injection. Inject parity error to Inbound on Completion Data 1 ram.
ERR EN RX CDQ 2	10	1b	RX CDQ 2 Parity Check Enable. If set to 1b enable Inbound on Completion Data 2 ram parity check.
ERR INJ RX CDQ 2 (SC)	11	0b	RX CDQ 2 ERR Injection. Inject parity error to Inbound on Completion Data 2 ram.
ERR EN RX CDQ 3	12	1b	RX CDQ 3 Parity Check Enable. If set to 1b enable Inbound on Completion Data 3 ram parity check.
ERR INJ RX CDQ 3 (SC)	13	0b	RX CDQ 3 ERR injection. Inject parity error to Inbound on Completion Data 3 ram.
Reserved	30:14	0x0	Reserved. Write 0x0, Ignore on read.
PCIe Flush	31	0b	PCIe Flush. When set, a device reset (*CTRL.DEV_RST*) also resets the PCIe Transaction Layer (TL) logic.

1. Bit loaded from NVM.

8.23.5 PCIe Parity Status Register - PCIEERRSTS (0x5BA8; RW1C)

Register logs uncorrectable parity errors detected in PCIe logic.

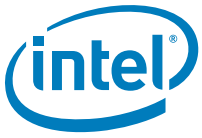
Reset by LAN_PWR_GOOD and PCIe reset.

Field	Bit(s)	Initial Value	Description
Reserved	2:0	0x0	Reserved. Write 0x0, ignore on read.
PAR ERR RX CDQ 0	3	0b	Rx CDQ 0 Parity Error. Parity error detection indication on Inbound on Completion Data 0 ram. Indicates detection of parity error in RAM if *PCIEERRCTL.ERR EN RX CDQ 0* is set. When set, stops all PCIe and DMA Rx and Tx activity from the function. To recover from this condition, the software device driver should issue a software reset by asserting *CTRL.DEV_RST* and re-initializing the port (refer to Section 7.7.1.1). ***Note:*** *PEIND.pcie_parity_fatal_ind* and *ICR.FER* interrupts are asserted if bits are not masked.
PAR ERR RX CDQ 1	4	0b	Rx CDQ 1 Parity Error. Parity error detection indication on Inbound on Completion Data 0 ram. Indicates detection of parity error in RAM if *PCIEERRCTL.ERR EN RX CDQ 1* is set. When set, stops all PCIe and DMA Rx and Tx activity from the function. To recover from this condition, the software device driver should issue a software reset by asserting *CTRL.DEV_RST* and re-initializing the port (refer to Section 7.7.1.1). ***Note:*** *PEIND.pcie_parity_fatal_ind* and *ICR.FER* interrupts are asserted if bits are not masked.
PAR ERR RX CDQ 2	5	0b	RX CDQ 2 Parity Error. Parity error detection indication on Inbound on Completion Data 2 ram. Indicates detection of parity error in RAM if *PCIEERRCTL.ERR EN RX CDQ 2* is set. When set, stops all PCIe and DMA Rx and Tx activity from the function. To recover from this condition, the software device driver should issue a software reset by asserting *CTRL.DEV_RST* and re-initializing the port (refer to Section 7.7.1.1). ***Note:*** *PEIND.pcie_parity_fatal_ind* and *ICR.FER* interrupts are asserted if bits are not masked.
PAR ERR RX CDQ 3	6	0b	RX CDQ 3 Parity Error. Parity error detection indication on Inbound on Completion Data 3 ram. Indicates detection of parity error in RAM if *PCIEERRCTL.ERR EN RX CDQ 3* is set. When set, stops all PCIe and DMA Rx and Tx activity from the function. To recover from this condition, the software device driver should issue a software reset by asserting *CTRL.DEV_RST* and re-initializing the port (refer to Section 7.7.1.1). ***Note:*** *PEIND.pcie_parity_fatal_ind* and *ICR.FER* interrupts are asserted if bits are not masked.
Reserved	31:7	0x0	Reserved. Write 0x0, ignore on read.

8.23.6 PCIe ECC Control Register - PCIEECCCTL (0x5BA4; RW)

Reset by LAN_PWR_GOOD.

Field	Bit(s)	Initial Value	Description
Reserved	11:0	0x511	Reserved.
ERR EN TX WR DATA	12	1b	Tx Write Request Data ECC Check Enable. If set to 1b enable outbound Posted Data ram ECC check.
ERR INJ TX WR DATA (SC)	13	0b	TX Write Request Data ECC ERR injection Inject ECC error to outbound Posted Data ram.



Field	Bit(s)	Initial Value	Description
ERR EN RETRY BUF	14	1b	Tx Retry Buffer ECC Check Enable. If set to 1b enable TX Retry Buffer ram ECC check.
ERR INJ RETRY BUF (SC)	15	0b	TX Retry Buffer1 ECC ERR injection Inject ECC error to TX Retry Buffer ram.
Reserved	31:16	0x0	Reserved. Write 0x0, Ignore on read.

8.23.7 PCIe ECC Status Register - PCIEECCSTS (0x5BAC; RW1C)

Note: Reset by LAN_PWR_GOOD and PCIe reset. Register logs occurrence of correctable ECC errors detected in PCIe logic.

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0	Reserved
ECC ERR TX WR DATA	4	0b	Tx Write Request Data ECC Correctable Error Correctable ECC error detection indication on Tx Write Request Data buffer ram.
ECC ERR RETRY BUF	5	0b	TX Retry Buffer ECC Correctable Error Correctable ECC error detection indication TX Retry Buffer ram.
Reserved	31:6	0x0	Reserved Write 0, ignore on read

8.23.8 PCIe ACL0 and ACL1 Register - PCIACL01 (0x5B7C; RW)

Reset by PCIe reset.

Field	Bit(s)	Initial Value	Description
ACL0	15:0	0x0	One of the four ACLs.
ACL1	31:16	0x0	One of the four ACLs.

8.23.9 PCIe ACL2 and ACL3 Register - PCIACL23 (0x5B80; RW to FW)

Field	Bit(s)	Initial Value	Description
ACL2	15:0	0	One of the four ACLs.
ACL3	31:16	0	One of the four ACLs.

8.23.10 LAN Port Parity Error Control Register - LANPERRCTL (0x5F54; RW)

Field	Bit(s)	Initial Value	Description
Reserved	8:0	0x0	Reserved. Write 0x0, ignore on read.
retx_buf_en	9	1b	Enable retx_buf parity error indication When set to 1b, enables the RETX buffer (re-transmit buffer) parity error detection and indication.
Reserved	31:10	0x0	Reserved. Write 0x0, ignore on read.

8.23.11 LAN Port Parity Error Inject Register - LANPERRINJ (0x5F5C; SC)

Field	Bit(s)	Initial Value	Description
Reserved	8:0	0b	Reserved. Write 0, ignore on read.
retx_buf_inj (SC)	9	0b	Inject retx_buf parity error
Reserved	31:10	0x0	Reserved. Write 0, ignore on read.

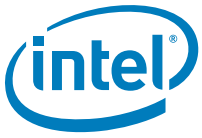
8.23.12 LAN Port Parity Error Status Register - LANPERRSTS (0x5F58; RW1C)

Field	Bit(s)	Initial Value	Description
Reserved	8:0	0x0	Reserved. Write 0x0, ignore on read.
retx_buf	9	0b	retx_buf Parity Error Indication. When set to 1b, indicates detection of parity error in the RETX buffer (re-transmit buffer) RAM if *LANPERRCTL.retx_buf_en* is set. When set, disables packet transmission. To recover from this condition, the software device driver should issue a software reset by asserting *CTRL.DEV_RST* and re-initializing the port. ***Note:*** *PEIND.lanport_parity_fatal_ind* and *ICR.FER* interrupts are asserted if bits are not masked.
Reserved	31:10	0x0	Reserved. Write 0x0, ignore on read.

8.23.13 Management Parity Configuration Register - MNGPARCTL (0x8F20; RW)

This register controls parity error reporting in Management block. Is reset by LAN_PWR_GOOD and Firmware reset.

Notes:



1. Register is Read-Only by Host and Read/Write by Management or NVM. Host write to this register is enabled only when the host-debug-write strap is active, otherwise, the register is read-only to the host driver.

Field	Bit(s)	Initial Value	Description
LAN-Par-En	0	1b	Parity Enable for LAN receive buffer (Management traffic from the LAN).
Host-Par-En	1	1b	Parity Enable for Host receive buffer (Management traffic from the host - OS2BMC).
Reserved	15:2	0x0	Reserved. Write 0, ignore on read.
Host-Par-Err-inj (SC)	17	0b	Parity Error injection to Host receive buffer (Receives OS2BMC traffic). ***Note:*** This bit is self cleared by HW after the next write to the buffer.
Reserved	30:18	0x0	Reserved. Write 0, ignore on read
MNG_HW_RST (SC)	31	0b	Management Hardware Reset. By setting this bit FW can reset all Management logic not related to ARC processor operation. ***Note:*** This bit is self cleared by HW following reset completion.

8.23.14 Management Parity Status Register - MNGPARSTS (0x8F24; RW1/C)

This register reports parity error occurrence in Management block. Is reset by LAN_PWR_GOOD and Firmware reset.

Notes:

1. Register is Read-Only by Host and Read/Write by Management or NVM. Host write to this register is allowed only when the host-debug-write strap is active; otherwise, the register is read-only to the host driver.
2. On parity error Firmware automatically generates a Firmware reset if the *PARITY_ERR_RST_EN* NVM bit is set.

Field	Bit(s)	Initial Value	Description
LAN-Par-Err	0	0b	LAN RX memory buffer parity error status indication. Indicates detection of parity error in LAN RX memory buffer (mng_pkt) if *MNGPARCTL.LAN-Par-En* bit is set. Will cause assertion of *PEIND.mng_parity_fatal_ind* and *ICR.FER* interrupt if bits are not masked. When set packet with parity error is dropped, RX path is disabled and an interrupt to internal ARC processor is generated. RX traffic can be restored either by issuing a FW reset or by asserting the *MNGPARCTL.MNG_HW_RST* bit. ***Note:*** FW clears bit by write 1b.
Host-Par-Err	1	0b	Host RX memory buffer parity error status indication. Indicates detection of parity error in Host (OS2BMC) RX memory buffer (mng_pkt) if *MNGPARCTL.Host-Par-En* bit is set. Will cause assertion of *PEIND.mng_parity_fatal_ind* and *ICR.FER* interrupt if bits are not masked. When set packet with parity error is dropped, RX path is disabled and an interrupt to internal ARC processor is generated. RX traffic can be restored either by issuing a FW reset or by asserting the *MNGPARCTL.MNG_HW_RST* bit. ***Note:*** FW clears bit by write 1b.
Reserved	31:2	0x0	Reserved. Write 0, ignore on read.

Note: If PARITY_ERR_RST_EN bit in NVM is set then FW reset is generated as a result of parity error occurrence (refer to Section 6.7.1.3).

8.23.15 Management ECC Configuration Register - MNGECCCTL (0x8F28; RW)

This register controls correctable ECC error reporting and ECC error injection in Management block. Is reset by LAN_PWR_GOOD and Firmware reset.

Notes:

1. Register is Read-Only by Host and Read/Write by Management or NVM. Host write to this register is allowed only when the host-debug-write strap is active; otherwise, the register is read-only to the host driver.
2. After power-up, FW should first initialize the RAMs and only after that enable ECC for the memories.

Field	Bit(s)	Initial Value	Description
ECC_EN_DRAM	0	1b	ECC enable for ARC Data ram (mng_data).
Reserved	7:1	0x0	Reserved. Write 0, Ignore on read.
DRAM-ECC-Inv1 (SC)	8	0b	ECC Error injection to ARC Data RAM. Invert bit 0 in the next written line. This bit is self cleared by HW after the next write to the memory.
Reserved	31:9	0x0	Reserved. Write 0, Ignore on read.

8.23.16 Management ECC Status Register - MNGECCSTS (0x8F2C; RW1/C)

This register reports correctable ECC error occurrence in Management block. Is reset by LAN_PWR_GOOD and Firmware reset.

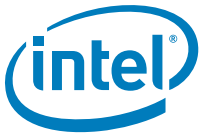
Notes:

1. Register is Read-Only by Host and Read/Write by Management or NVM. Host write to this register is allowed only when the host-debug-write strap is active; otherwise, the register is read-only to the host driver.

Field	Bit(s)	Initial Value	Description
ECC_ERR_DRAM	0	0x0	ARC Data ram (mng_data) correctable ECC occurrence status indication.
Reserved	31:1	0x0	Reserved. Write 0, Ignore on read.

8.23.17 DMA Receive Parity and ECC Control - DRPARC (0x3F04; RW/SC)

Field	Bit(s)	Initial value	Description
RAM_DRX_DESC_ECC_EN	0	1b	ram_drx_desc ECC check enable.ram_drx_desc_ecc_en (memory in the DP for descriptors from the DH) ECC check enable.



Field	Bit(s)	Initial value	Description
RAM_DRX_DESC_INJ_ERR	1	0b	ram_drx_desc ECC error inject.ram_drx_desc_ecc_en (memory in the DP for descriptors from the DH) ECC error inject. Written on Rx packet reception.
RAM_DRX_DHRF_PAR_EN	2	1b	ram_drx_dhrf parity check enable.ram_drx_dhrf (drx_dh memory in Descriptor Handler responsible for keeping all the per queue registers) parity check enable.
RAM_DRX_DHRF_INJ_ERR	3	0b	ram_drx_dhrf parity error inject.ram_drx_dhrf (drx_dh memory in Descriptor Handler responsible for keeping all the per queue registers) parity error inject. Written with every register write or status change in the DH. Bit 0 is written when a queue is enabled.
RAM_DRX_ICACHE_ECC_EN	4	1b	ram_drx_icache ECC check enable.ram_drx_icache (drbuf internal descriptor cache) ECC check enable.
RAM_DRX_ICACHE_INJ_ERR	5	0b	ram_drx_icache ECC error inject.ram_drx_icache (drbuf internal descriptor cache) ECC error inject.This RAM is written on Rx packet reception.
RAM_DRX_SRRCTL_ECC_EN	6	1b	ram_drx_srrctl ECC check enable.ram_drx_srrctl (drx_pb RAM for CSRs in the DBU Rx) ECC check enable.
RAM_DRX_SRRCTL_INJ_ERR	7	0b	ram_drx_srrctl ECC error inject.ram_drx_srrctl (drx_pb RAM for CSRs in the DBU Rx) ECC check enable.This RAM is written on SRRCTL or RDT write, or on Rx packet reception.
RESERVED	31:8	0x0	Reserved.Write 0, ignore on read.

8.24 Power Management Register Description

The following registers are used to control various power saving features.

8.24.1 DMA Coalescing Management Threshold - DMCMNGTH (0x8F30;RW)

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0b	Reserved. Write 0x0, ignore on read.
DMCMNGTHR	19:4	0x100	DMA Coalescing Management Threshold. This value defines the DMA coalescing management threshold in 16 byte units. When the amount of empty space in the internal transmit buffer exceeds the DMCMNGTHR value, DMA coalescing is stopped and PCIe moves to an L0 state. ***Note:*** If this value is 0x0, a condition to move out of DMA coalescing due to the passing of the DMA coalescing management threshold level is disabled. Under some conditions, there can be a deviation of up to 16-bytes from the value written in this field. (HSD 359146)
Reserved	31:20	0b	Reserved. Write 0x0, ignore on read.

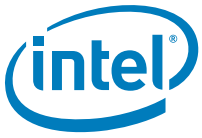
8.24.2 Latency Tolerance Reporting (LTR) Minimum Values - LTRMINV (0x5BB0; RW)

Register reset by PCIe power Good reset, FLR reset and move from D3 to D0.

Field	Bit(s)	Initial Value	Description
LTRV	9:0	0x5	Latency Tolerance Value. This field indicates the latency tolerance supported when conditions for minimum latency tolerance exist (Refer to Section 5.9.2.1). *LTRV* values are multiplied by 32,768 ns or 1,024 ns depending on the *Scale* field, to indicate latency tolerance supported in nanoseconds. A value of 0 indicates that the device is impacted by any delay and that best possible service is requested. Foxville reports the same value for both snoop and no snoop requirements. If no memory latency requirement exists for either snoop or no snoop accesses, the appropriate *Requirement* bit is cleared. ***Note:*** Software should subtract time required to move from L1 to L0 from LTR value.
Scale	12:10	011b	Latency Scale. This field provides a scale for the value contained within the *LTRMINV.LTRV* field. Encoding: 010b = *LTRV* value times 1,024 ns. 011b = *LTRV* value times 32,768 ns. Others = Reserved.
Reserved	14:13	0x0	Reserved. Write 0x0, ignore on read.
LSNP Requirement	15	0b	LTR Snoop Requirement. 0b = No latency requirements in snoop memory access. 1b = Latency tolerance in snoop memory access specified in *LTRMINV.LTRV* field.
Reserved	30:16	0x0	Reserved. Write 0x0, ignore on read.
LNSNP Requirement	31	0b	LTR Non-snoop Requirement. 0b = No latency requirements in non-snoop memory access. 1b = Latency tolerance in non-snoop memory access specified in *LTRMINV.LTRV* field.

8.24.3 Latency Tolerance Reporting (LTR) Maximum Values - LTRMAXV (0x5BB4; RW)

Register reset by PCIe power Good reset, FLR reset and move from D3 to D0.



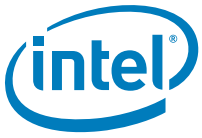
Field	Bit(s)	Initial Value	Description
LTRV	9:0	0x5	Latency Tolerance Value. This field indicates the latency tolerance supported when conditions for maximum latency tolerance exist (Refer to Section 5.9.2.2). *LTRV* values are multiplied by 32,768 ns or 1,024 ns depending on the *Scale* field to indicate latency tolerance supported in nanoseconds. A value of 0 indicates that the device is impacted by any delay and that the best possible service is requested. Foxville reports the same value for both snoop and no snoop requirements. If no memory latency requirement exists for either snoop or no snoop, accesses the appropriate *Requirement* bit is cleared. ***Note:*** Software should subtract the time required to move from L1 to L0 from LTR value.
Scale	12:10	011b	Latency Scale. This field provides a scale for the value contained within the *LTRMAXV.LTRV* field. Encoding: 010b = *LTRV* value times 1,024 ns. 011b = *LTRV* value times 32,768 ns. Others = Reserved.
Reserved	14:13	0x0	Reserved. Write 0x0, ignore on read.
LSNP Requirement	15	0b	LTR Snoop requirement 0b = No latency requirements in snoop memory access. 1b = Latency tolerance in snoop memory access specified in *LTRMAXV.LTRV* field.
Reserved	30:16	0x0	Reserved. Write 0x0, ignore on read.
LNSNP Requirement	31	0b	LTR Non-snoop Requirement. 0b = No latency requirements in non-snoop memory access. 1b = Latency tolerance in non-snoop memory access specified in *LTRMAXV.LTRV* field.

8.24.4 Latency Tolerance Reporting (LTR) Control - LTRC (0x01A0; RW)

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved. Write 0b, ignore on read.
LTR_MIN	1	0b	LTR Send Minimum Values. When set to 1b, Foxville sends a PCIe LTR message with the LTR snoop value, LTR no-snoop value and LTR requirement bits as defined in the LTRMINV register. ***Notes:*** 1. To resend a LTR message with the minimum value defined in the LTRMINV register, this bit should be cleared and set again. 2. LTR_MIN and LTR_MAX bits are exclusive. 3. A new PCIe LTR message is sent only if the last PCIe LTR message sent had a latency tolerance value different then the value specified in the LTRMINV register.
LTR_MAX	2	0b	LTR Send Maximum Values. When set to 1b, Foxville sends a PCIe LTR message with the LTR snoop value, LTR no-snoop value and LTR requirement bits as defined in the LTRMAXV register. ***Notes:*** 1. To resend a LTR message with the maximum value defined in the LTRMAXV register, this bit should be cleared and set again. 2. LTR_MIN and LTR_MAX bits are exclusive. 3. A new PCIe LTR message is sent only if the last PCIe LTR message sent had a latency tolerance value different then the value specified in the LTRMAXV register.
PDLS_EN	3	1b	Port Disable LTR Send Enable. 0b = Do not issue a PCIe LTR message with requirement bits cleared on port disable (Rx and Tx disabled). 1b = Issue a PCIe LTR message with requirement bits cleared on port disable (Rx and Tx disabled).
LNKDLS_EN	4	1b	Link Disconnect LTR Send Enable. 0b = Do not issue a PCIe LTR message with requirement bits cleared on link disconnect. 1b = Issue a PCIe LTR message with requirement bits cleared on link disconnect.
EEEMS_EN	5	0b	EEE LPI LTR Max Send Enable. When this bit is set and link is in a Rx EEE LPI (Low Power Idle) state, Foxville sends a PCIe LTR message with the LTR snoop value, LTR no-snoop value and LTR requirement bits as defined in the LTRMAXV register. For further information, refer to Section 5.9. 0b = Do not issue a PCIe LTR messages with the LTRMAXV value as a result of Rx link entering EEE LPI state. 1b=Issue PCIe LTR messages with a LTRMAXV value as a result of Rx link entering EEE LPI state. ***Note:*** This bit is reset to 0b by hardware following link disconnect to enable software to re-negotiate Tw_system time and update the LTRMAXV value.
Reserved	31:6	0x0	Reserved. Write 0x0, ignore on read.

8.24.5 Energy Efficient Ethernet (EEE) Register - EEER (0x0E30; RW)

Register reset on LAN_PWR_GOOD.



Field	Bit(s)	Initial Value	Description
Tw_system	15:0	0x0	Time expressed in microseconds that no data is transmitted following a move from the EEE TX LPI link state to a link active state. This field holds the transmit Tw_sys_tx value negotiated during EEE LLDP negotiation. ***Notes:*** 1. If this value is lower than the minimum Tw_sys_tx value defined in IEEE802.3az clause 78.5 (30 µs for 100BASE-TX and 16.5 µs for 1000BASE-T and 29.5 us for 2500BASE-T) then the interval where no data is transmitted following a move out of the EEE TX LPI state defaults to a minimum Tw_sys_tx as defined in *EEE_SU* and *EEE_SU_2P5* registers. 2. Following a link disconnect or auto-negotiation the value of this field returns to its default value until software re-negotiates a new tw_sys_tx value via EEE LLDP. ***Note:*** When transmitting flow control frames, Foxville waits the minimum time defined in the IEEE802.3az standard (as defined in the *EEE_SU* and *EEE_SU_2P5* registers) before transmitting a flow control packet. Foxville does not wait the Tw_system time following an exit of LPI before transmitting a flow control frame.
TX_LPI_EN	16	0b[1]	Enable Entry into EEE LPI on Tx Path. 0b = Disable entry into EEE LPI on Tx path. 1b = Enable entry into EEE LPI on Tx path. Refer to Section 3.6.6.1 for additional information on EEE Tx LPI entry. ***Notes:*** 1. Even when *TX_LPI_EN* is set to 1b, Foxville will not enable entry into the Tx LPI state for at least 1 second (depending on *EEE_SU.TX_LU_LPI_DLY* field) following the change of link_status to OK as defined in IEEE802.3az clause 78.1.2.1. 2. Even if the *TX_LPI_EN* bit is set, Foxville initiates entry into the Tx EEE LPI link state only if EEE support at the link speed was negotiated during auto-negotiation. 3. Foxville will enter EEE TX LPI only after conditions exist for a duration defined in the *EEE_SU.TEEE_DLY* field.
RX_LPI_EN	17	1b	Enable Entry into EEE LPI on Rx Path 0b = Disable entry into EEE LPI on Rx path. 1b = Enable entry into EEE LPI on Rx path. ***Notes:*** 1. Even if the *RX_LPI_EN* bit is set, Foxville recognizes entry into Rx EEE LPI link state only if EEE support at the link speed was negotiated during auto-negotiation. 2. When set and link moves into Rx LPI, a LTR message with the value defined in the LTRMAXV register is sent on the PCIe, if *LTRC.EEEMS_EN* is set.
LPI_FC	18	1b	Enable EEE Tx LPI Entry on Flow Control. Enable EEE Tx LPI state entry when the link partner sent a PAUSE flow control frame, even if the internal transmit buffer is not empty, transmit descriptors are available or management traffic is pending. ***Notes:*** 1. Foxville enters the Tx LPI state when no data is transmitted and not in mid-packet. 2. Entry into Tx LPI on flow control is enabled only if either *EEER.TX_LPI_EN* is set to 1b or *EEER.Force_TLPI* is set to 1b. 3. Receiving XON frame causes a move out of LPI if a transmit is pending.
Force_TLPI	19	0b	Force Tx LPI. When set, the PHY is forced into the EEE Tx LPI state if there is no Tx management traffic. ***Notes:*** 1. Foxville enters the Tx LPI state when no data is transmitted and not in mid-packet. 2. When set, Foxville enters Tx LPI even if *EEER.TX_LPI_EN* is set to 0b. 3. To work in Force TX LPI mode, *EEE_SU.TEEE_DLY* field should have a value greater then 0.

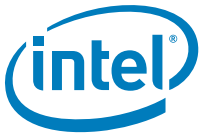
Field	Bit(s)	Initial Value	Description
Reserved	20	0x0	Reserved. Write 0x0, ignore on read.
Reserved	27:21	0x0	Reserved. Write 0x0, ignore on read.
EEE_FRC_AN	28	0b	Force EEE Auto-negotiation. When this bit is set to 1b,it enables EEE operation in the internal MAC logic even if the link partner does not support EEE. Should be set to 1b to enable testing of EEE operation via MAC loopback (refer to y).
EEE NEG (RO)	29	X	EEE Support Negotiated on Link. 0b = EEE operation not supported on link. 1b = EEE operation supported on link. Note: Status reported by this bit shall be ignored when the port is operated in half duplex mode.
RX LPI Status (RO)	30	X	Rx Link in LPI State. 0b = Rx in active state. 1b = Rx in LPI state.
TX LPI Status (RO)	31	X	Tx Link in LPI State. 0b = Tx in active state. 1b = Tx in LPI state.

1. Loaded from NVM.

8.24.6 Energy Efficient Ethernet (EEE) Setup Register - EEE_SU (0x0E34; RW)

Register reset on LAN_PWR_GOOD.

Field	Bit(s)	Initial Value	Description
Tw_min_1000	7:0	0x21	Minimum time expressed in 0.5 microseconds for 1000BASE-T operation, where no data will be transmitted following move out of EEE LPI TX state. Time should be equal or greater than minimum Tw_sys_tx value defined in IEEE802.3az clause 78.5 for 1000BASE-T (16.5 μsec). ***Note:*** The idle time value defined in this field, is used when moving out of EEE TX LPI state to transmit flow control frames even if value specified in *EEER.Tw_system* field is higher.
Tw_min_100	15:8	0x3C	Minimum time expressed in 0.5 microseconds for 100BASE-TX operation, where no data will be transmitted following move out of EEE LPI TX state. Time should be equal or greater than minimum Tw_sys_tx value defined in IEEE802.3az clause 78.5 for 100BASE-TX (30 μsec). ***Note:*** The idle time value defined in this field, is used when moving out of EEE TX LPI state to transmit flow control frames even if value specified in *EEER.Tw_system* field is higher.
Tw_wake_min	21:16	0x0	Minimum time expressed in 10 microseconds, between sending a request to move into EEE TX LPI and sending a request to move back to active state. Refer to Section 3.6.6.2 for additional information on conditions to exit TX LPI. ***Note:*** If conditions to exit LPI during the Tw_wake_min interval cease to exist than Foxville will not move out of TX LPI after timer has expired.
XXMII_LPI_Stop	22	1b	Stop MII/GMII clock on internal MII/GMII interface when moving to LPI state as defined in IEEE802.3az. 0b - Do not stop XXMII clock. 1b - Stop XXMII clock.



Field	Bit(s)	Initial Value	Description
LPI_Clock_Stop	23	1b	Stop clocks when TX and RX in LPI mode. 0b - Do not stop clocks when RX and TX in LPI mode. ***Note:*** 1b - Stop clocks when RX and TX in LPI mode.
TX_LU_LPI_DLY	25:24	11b	Delay to enable entry of TX EEE LPI state following Link-up indication. 00b - No delay 01b - 100 µs 10b - 1 mS 11b - 1 Second (the only permitted value at 2.5G) Refer to Section 3.6.6.1 for additional information on EEE TX LPI entry. ***Note:*** IEEE802.3az clause 78.1.2.1 defines delay of 1 second following link-up.
TEEE_DLY	31:26	0x2	TX EEE LPI Entry delay Field defines delay to EEE entry once conditions to enter EEE LPI are detected. Field resolution is 20 µs. Refer to Section 3.6.6.1 for additional information on EEE TX LPI entry. ***Note:*** If conditions to enter LPI during the *TEEE_DLY* interval cease to exist. Foxville will not enter TX LPI and continue normal operation.

8.24.7 Energy Efficient Ethernet (EEE) 2.5Gb/s Setup Register - EEE_SU_2P5 (0x0E3C; RW)

Register reset on LAN_PWR_GOOD.

Field	Bit(s)	Initial Value	Description
TW_MIN_2500	7:0	0x3B	Minimum time expressed in 0.5 microseconds for 2500BASE-T operation, where no data will be transmitted following move out of EEE LPI TX state. Time should be equal or greater than minimum Tw_sys_tx value defined in IEEE802.3bz (29.44 uSec). The idle time value defined in this field, is used when moving out of EEE TX LPI state to transmit flow control frames even if value specified in EEER.Tw_system field is higher.
RESERVED	31:8	0x0	

8.25 PHY Software Interface

8.25.1 Internal PHY Configuration - IPCNFG (0x0E38, RW)

Register reset by LAN_PWR_GOOD only.

Field	Bit(s)	Initial Value	Description
Enable Automatic Crossover	0	1b	When set, the device automatically determines whether or not it needs to cross over between pairs so that an external cross-over cable is not required. This bit is loaded from bit 9 in the Initialization Control 3 NVM word.
Reserved	1	0b	Reserved
EEE_100M_AN	2	1b	Report EEE 100 Mb/s Capability in Auto-negotiation 0b = Do not report EEE 100 Mb/s capability in auto-negotiation. 1b = Report EEE 100 Mb/s capability in auto-negotiation. Changing the value of this bit causes link re-negotiation. This bit is loaded from bit 0 in the Initialization Control 2 NVM word.

Field	Bit(s)	Initial Value	Description
EEE_1G_AN	3	1b	Report EEE 1 GbE Capability in Auto-negotiation. 0b = Do not report EEE 1 GbE capability in auto-negotiation. 1b = Report EEE 1 GbE capability in auto-negotiation. Changing the value of this bit causes link re-negotiation. This bit is loaded from bit 1 in the Initialization Control 2 NVM word.
EEE_2_5G_AN	4	1b	Report EEE 2.5 GbE Capability in Auto-negotiation. 0b = Do not report EEE 2.5 GbE capability in auto-negotiation. 1b = Report EEE 2.5 GbE capability in auto-negotiation. Changing the value of this bit causes link re-negotiation. This bit is loaded from bit 2 in the Initialization Control 2 NVM word.
RSVD	31:5	0x0	Reserved

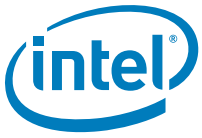
8.25.2 PHY Power Management - PHPM (0x0E14, RW)

Register reset by LAN_PWR_GOOD only.

The PHPM register controls the internal PHY power management operation.

Please note that the PHPM is a shared resource accessed by both the software and the embedded firmware. As such, before accessing this register, both the software and the firmware should comply with the "Acquire Ownership Over a Shared Resource" flow described in Section 4.8.1. This shared resource is indicated by the MAC_CSR_SM flag in the SW_FW_SYNC semaphore registers.

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved
Restart_AutoNeg	1	0b	Restart Auto Negotiation. Any change in this bit causes the device to restart auto negotiation. This bit is reflected to the "Restart Auto-negotiation" signal to the PHY.
RSVD	2	0b	Reserved
Disable 1000 in non-D0a	3	1b	Disables 1000 Mb/s operation in non-D0a states. This bit is loaded from the Dis1000_nonD0a flag in the NVM.
Link Energy Detect (RO Status flag)	4	0b	The Link Energy Detect is set when the PHY detects energy on the link.
Go Link disconnect	5	1b	Setting this bit causes the PHY to enter link disconnect immediately.
Disable 1000	6	0b	When the Disable 1000 bit is set, 1000 Mb/s is disabled in all power states. This bit is loaded from the Dis1000 flag in the NVM.
SPD_B2B_EN	7	1b	This flag enables the Smart Power Down Back-to-Back. The setting of this flag is captured in the always on power rail (used during the SPD state).
RST_COMPL (RO Status flag)	8	0b	It is set to '1' at the completion of the internal PHY reset sequence and cleared on read.
Disable 100 in non-D0a	9	0b	Disables 100 Mb/s operation in non-D0a states. This bit is loaded from the "Disable 100 in non-D0a" flag in the NVM.
ULP_Req	10	0b	Setting this bit to '1' by the firmware transits the device to the Ultra Low Power State if all conditions to enter the ULP are met. This bit is RO to the software.
Disable 2500	11	0b	When set, disables 2500 Mb/s in all power modes. This bit is loaded from the Dis2500 flag in the NVM.
Disable 2500 in non-D0a	12	0b	Disables 2500 Mb/s operation in non-D0a states. This bit is loaded from the Dis1000_nonD0a flag in the NVM.
ULP_Trig (RO)	13	0b	This flag is RO, set by the device. When set to '1' Foxville transits to the Ultra Low Power State after a ULP_Delay.



Field	Bit(s)	Initial Value	Description
ULP_Delay	15:14	01b	Programmable delay entering the ULP in msec units.
Link_Energy_En	16	0b	Enable Link Energy for the ULP state
Dev_Off_En	17	0b	Enable LAN_DISABLE_N for the ULP state
Dev_Off_State (RO)	18	X	Reflect the level of the LAN_DISABLE_N signal
ULP_En (RO)	19	0b	This flag is RO, set by the device tracking the ULP_Trig flag after the programmable ULP_Delay. At '1' Foxville transits to the ULP State.
RSVD	31:20	0x0	Reserved

8.26 PHY MDIO Registers

The PHY Registers are accessible via the MDIC register. This section list the IEEE 802.3 standard management registers and the PHY specific registers. All other PHY registers as well as its detailed description are presented in a separate GPY211 PHY data sheet.

Table 8-1 PHY MDIO Registers

Offset	Abbreviation	Name	Init Value
Standard Management Registers			
0x0	STD_CTRL	Control	0x9040
0x1	STD_STAT	Status	0x7949
0x2	STD_PHYID1	PHY Identifier 1	0x67C9
0x3	STD_PHYID2	PHY Identifier 2	0xDC00
0x4	STD_AN_ADV	Auto-Negotiation Advertisement	0x01E1
0x5	STD_AN_LPA	Auto-Negotiation Link-Partner Ability	0x0000
0x6	STD_AN_EXP	Auto-Negotiation Expansion	0x0004
0x7	STD_AN_NPTX	Auto-Negotiation Next-Page Transmit	0x2001
0x8	STD_AN_NPRX	Auto-Negotiation Link-Partner Received Next Page	0x2001
0x9	STD_GCTRL	Gigabit Control	0x0300
0xA	STD_GSTAT	Gigabit Status	0x0000
0xB	STD_RES11	Reserved	0x0000
0xC	STD_RES12	Reserved	0x0000
0xD	STD_MMDCTRL	MMD Access Control	0x0000
0xE	STD_MMDDATA	MMD Access Data	0x0000
0xF	STD_XSTAT	Extended Status	0x3000
PHY-Specific Management Registers			
0x10	PHY_PERF	Physical Layer Performance Status	0x80FF
0x11	PHY_STAT1	Physical Layer Status 1	0x0000
0x12	PHY_STAT2	Physical Layer Status 2	0x0000
0x13	PHY_CTL1	Physical Layer Control 1	0x0001
0x14	PHY_CTL2	Physical Layer Control 2	0x8006
0x15	PHY_ERRCNT	Error Counter	0x0000
0x16	PHY_RES22	Reserved	0x0000
0x17	PHY_RES23	Reserved	0x0000
0x18	PHY_MIISTAT	Media-Independent Interface Status	0x0000
0x19	PHY_IMASK	Interrupt Mask Register	0x0000
0x1A	PHY_ISTAT	Interrupt Status Register	0x0000
0x1B	PHY_LED	LED Control Register	0x0F00
0x1C	PHY_TPGCTRL	Test-Packet Generator Control	0x0000

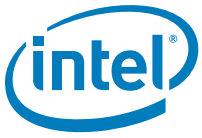


Table 8-1 PHY MDIO Registers

Offset	Abbreviation	Name	Init Value
0x1D	PHY_TPGDATA	Test-Packet Generator Data	0x00AA
0x1E	PHY_FWV	Firmware Version Register	0x8304
0x1F	PHY_RES1F	Reserved	0x0000

9.0 PCIe Programming Interface

9.1 PCIe* Compatibility

PCIe is completely compatible with existing deployed PCI software. Foxville contain the following regions of the PCI configuration space:

- Mandatory PCI configuration registers
- Power management capabilities
- MSI and MSI-X capabilities
- PCIe extended capabilities

9.2 PCIe Register Map

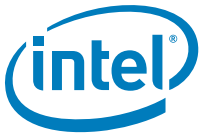
9.2.1 Register Attributes

Configuration registers are assigned one of the attributes described in the following table.

Table 9-1. Configuration Registers

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.
R/W1C	Read-only status, write-1-to-clear status register, writing a 0b to R/W1C bits has no effect.
ROS	Read-only register with sticky bits: Register bits are read-only and cannot be altered by software. Bits are not cleared by reset and can only be reset with the PWRGOOD signal. Devices that consume AUX power are not allowed to reset sticky bits when AUX power consumption (either via AUX power or PME enable) is enabled.
RWS	Read-write register: Register bits are read-write and can be either set or reset by software to the desired state. Bits are not cleared by reset and can only be reset with the PWRGOOD signal. Devices that consume AUX power are not allowed to reset sticky bits when AUX power consumption (either via AUX power or PME enable) is enabled.
R/W1CS	Read-only status, write-1-to-clear status register: Register bits indicate status when read, a set bit indicating a status event can be cleared by writing a 1b. Writing a 0b to R/W1C bits has no effect. Bits are not cleared by reset and can only be reset with the PWRGOOD signal. Devices that consume AUX power are not allowed to reset sticky bits when AUX power consumption (either via AUX power or PME enable) is enabled.
HwInit	Hardware initialized: Register bits are initialized by firmware or hardware mechanisms such as pin straping or serial NVM. Bits are read-only after initialization and can only be reset (for write-once by firmware) with PWRGOOD signal.
RsvdP	Reserved and preserved: Reserved for future R/W implementations; software must preserve value read for writes to bits.
RsvdZ	Reserved and zero: Reserved for future R/W1C implementations; software must use 0b for writes to bits.

The PCI configuration registers map is listed in Table 9-2. Refer to a detailed description for registers loaded from the NVM at initialization time. Note that initialization values of the configuration registers are marked in parenthesis.



9.2.2 PCIe Configuration Space Summary

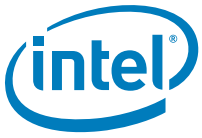
Table 9-2. PCIe Configuration Registers Map

Section	Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
Mandatory PCI register	0x0	Device ID		Vendor ID	
	0x4	Status Register		Control Register	
	0x8	Class Code (0x020000/0x010000)			Revision ID
	0xC	BIST (0x00)	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 Device 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)
Power management capability	0x40	Power Management Capabilities		Next Pointer (0x50)	Capability ID (0x01)
	0x44	Data	Bridge Support Extensions	Power Management Control & Status	
MSI capability	0x50	Message Control (0x0180)		Next Pointer (0x70)	Capability ID (0x05)
	0x54	Message Address			
	0x58	Message Upper Address			
	0x5C	Reserved		Message Data	
	0x60	Mask bits			
	0x64	Pending bits			
MSI-X capability	0x70	Message Control (0x0004)		Next Pointer (0xA0)	Capability ID (0x11)
	0x74	Table Offset			
	0x78	PBA offset			
CSR Access Registers	0x98	IOADDR			
	0x9C	IODATA			

Table 9-2. PCIe Configuration Registers Map

Section	Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
PCIe capability	0xA0	PCIe Capability Register (0x0002)		Next Pointer (0xE0)	Capability ID (0x10)
	0xA4	Device Capability			
	0xA8	Device Status		Device Control	
	0xAC	Link Capabilities			
	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	Reserved			
	0xD0	Link Status 2		Link Control 2	
	0xD4	Reserved			
	0xD8	Reserved		Reserved	
VPD capability	0xE0	VPD address		Next Pointer (0x00)	Capability ID (0x03)
	0xE4	VPD data			
AER capability	0x100	Next Capability Ptr: 12 bit field (0x140)	Version (0x2): 4 bit field	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			
Serial ID capability	0x140	Next Capability Ptr: 12 bit field (0x1C0)	Version (0x1): 4 bit field	Serial ID Capability ID (0x0003)	
	0x144	Serial Number Register (Lower Dword)			
	0x148	Serial Number Register (Upper Dword)			
LTR capability	0x1C0	Next Capability Ptr: 12 bit field (0x1F0)	Version (0x1): 4 bit field	LTR Capability ID (0x18)	
	0x1C4	Maximum Non-Snooped Platform Latency Tolerance Register		Maximum Snooped Platform Latency Tolerance Register	
L1 Sub States	0x1E0	Next Capability Ptr: 12 bit field (0x000)	Version (0x1): 4 bit field	L1 Sub States Capability ID (0x1E)	
	0x1E4	L1 PM Sub states Capabilities Register			
	0x1E8	L1 PM Sub states Control 1 Register			
	0x1EC	L1 PM Sub states Control 2 Register			
Precision Time Measurement	0x1F0	Next Capability Ptr: 12 bit field (0x1E0)	Version (0x1): 4 bit field	Precision Time Measurement ID (0x18)	
	0x1F4	PTM Capability Register			
	0x1F8	PTM Control Register			

A description of the registers is provided in the following sections.



9.3 Mandatory PCI Configuration Registers

9.3.1 Vendor ID (0x0; RO)

This value can be loaded automatically from NVM address 0x0E at power up or reset. A value of 0x8086 is the default for this field at power up if the NVM does not respond or is not programmed.

Note: To avoid a system hang situation, if a value of 0xFFFF is read from the NVM, the value of the Vendor ID field defaults back to 0x8086.

9.3.2 Device ID (0x2; RO)

This is a read-only register. This field identifies individual Foxville functions. It can be auto-loaded from the NVM during initialization with a different value. The following table lists the possible values according to the SKU and functionality.

NVM Address	Device ID	SKU / NVM	Description
0x0D	0x15FD	Blank NVM	Client Foxville with Empty Flash Image
	0x15F2	Foxville LM	Client 2.5G LAN vPro / Corporate
	0x15F3	Foxville V	Client 2.5G LAN Consumer
	0x15F7	Foxville 1G	Client 1G only Foxville
	0x15F8	Foxville I	Client 2.5G Industrial Temp

9.3.3 Command Register (0x4; R/W)

This is a read/write register.

Bit(s)	R/W	Initial Value	Description
0	R/W[1]	0b	I/O Access Enable
1	R/W	0b	Memory Access Enable
2	R/W	0b	Bus Master Enable (BME)
3	RO	0b	Special Cycle Monitoring Hardwired to 0b.
4	RO	0b	MWI Enable Hardwired to 0b.
5	RO	0b	Palette Snoop Enable Hardwired to 0b.
6	RW	0b	Parity Error Response
7	RO	0b	Wait Cycle Enable Hardwired to 0b.
8	RW	0b	SERR# Enable
9	RO	0b	Fast Back-to-Back Enable
10	RW	0b	Interrupt Disable[2]
15:11	RO	0x0	Reserved

1. If IO_Sup bit in PCIe Init Configuration 2 NVM Word (0x19) is 0, I/O Access Enable bit is RO with a value of 0.

2. The Interrupt Disable register bit is a read-write bit that controls the ability of a PCIe device to generate a legacy interrupt message. When set, devices are prevented from generating legacy interrupt messages.

9.3.4 Status Register (0x6; RO)

Bits	R/W	Initial Value	Description
2:0		000b	Reserved
3	RO	0b	Interrupt Status[1]
4	RO	1b	New Capabilities Indicates that a device implements extended capabilities. Foxville sets this bit, and implements a capabilities list, to indicate that it supports PCI power management, Message Signaled Interrupts (MSI), Enhanced Message Signaled Interrupts (MSI-X), Vital Product Data (VPD), and the PCIe extensions.
5		0b	66 MHz Capable Hardwired to 0b.
6		0b	Reserved
7		0b	Fast Back-to-Back Capable Hardwired to 0b.
8	R/W1C	0b	Data Parity Reported
10:9		00b	DEVSEL Timing Hardwired to 0b.
11	R/W1C	0b	Signaled Target Abort
12	R/W1C	0b	Received Target Abort
13	R/W1C	0b	Received Master Abort
14	R/W1C	0b	Signaled System Error
15	R/W1C	0b	Detected Parity Error

1. The *Interrupt Status* field is a RO field that indicates that an interrupt message is pending internally to the device.

9.3.5 Revision (0x8; RO)

The default revision ID for Foxville A step is 0x00 and for the B step it is 0x01 (as programmed in the "Step REV ID" field in the MREVID register). The value of the rev ID field in the PCIe configuration space is a logic XOR between the above default values and the DEVREVID field in the NVM (at word 0x1E).

9.3.6 Class Code (0x9; RO)

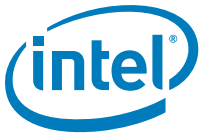
The class code is a RO hard coded value that identifies Foxville's functionality.

- 0x020000/0x010000 - Ethernet/SCSI Adapter[1]

9.3.7 Cache Line Size (0xC; R/W)

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. Field is loaded from the *PCIe Init Configuration 3* (Word 0x1A) NVM word and defines cache line size in Dwords. In systems, the value is 0x10.

1. Selected according to bit 11 in *Device Rev ID* NVM word.



9.3.8 Latency Timer (0xD; RO)

Not used. Hardwired to zero.

9.3.9 Header Type (0xE; RO)

This indicates if a device is single function or multifunction. If a single LAN function is the only active one then this field has a value of 0x00 to indicate a single function device.

9.3.10 BIST (0xF; RO)

BIST is not supported in Foxville.

Bit(s)	Read/ Write	Initial Value	Description
7:0	R	0	Reserved

9.3.11 Base Address Registers (0x10...0x27; R/W)

The Base Address registers (BARs) are used to map Foxville register space. Foxville has a memory BAR, IO BAR and MSI-X BAR described in Table 9-3 below. The BARs location and sizes are described in Table 9-4 and Table 9-5. The fields within each BAR are then described in Table 9-6. 32-bit addresses may be used in one register for each memory mapping window or 64-bit addresses with 2 registers for each memory mapping window depending on NVM setting.

Table 9-3. Base Address Registers Description -

Mapping Windows	Mapping Description
Memory BAR	The internal registers memories and external Flash device are accessed as direct memory mapped offsets from the Base Address register. Software can access a Dword or 64 bits. The Flash space in this BAR is enabled by the "Flash Size bar" field in the NVM. Address 0 in the Flash device is mapped to address 128K in the Memory BAR. When the usable Flash size + CSR space is smaller than the memory BAR, then accessing addresses above the top of the Flash wraps back to the beginning of the Flash.
IO BAR	All internal registers and memories can be accessed using I/O operations. There are two 4-byte registers in the IO mapping window: Addr Reg and Data Reg accessible as Dword entities. I/O BAR support depends on the *IO_Sup* bit in the NVM "*PCIe Init Configuration 2*" word.
MSI-X BAR	The MSI-X vectors and Pending bit array (PBA) structures are accessed as direct memory mapped offsets from the MSI-X BAR. Software can access Dword entities.

9.3.11.1 32-bit LAN BARs Mode Mapping

This mapping is selected when bit 10 in the *Functions Control* NVM word is equal to 1b.

Table 9-4. Base Address Setting in 32bit BARs Mode (NVM.BAR32 = 1b)

BAR	Addr	31 5	4	3	2	1	0
0	0x10	Memory CSR + FLASH BAR (R/W - 31:17; RO - 16:4 (0x0))		0/1	0	0	0
1	0x14	Reserved (read as all 0b's)					
2	0x18	IO BAR (R/W - 31:5)	0	0	0	0	1/0

Table 9-4. Base Address Setting in 32bit BARs Mode (NVM.BAR32 = 1b)

BAR	Addr	31 5	4	3	2	1	0
3	0x1C	MSI-X BAR (R/W - 31:14; RO - 13:4 (0x0))		0/1	0	0	0
4	0x20	Reserved (read as all 0b's)					
5	0x24	Reserved (read as all 0b's)					

9.3.11.2 64-bit LAN BARs Mode Mapping

This mapping is selected when bit 10 in the *Functions Control* NVM word is equal to 0b.

Table 9-5. Base Address Setting in 64bit BARs Mode (NVM.BAR32 = 1b)

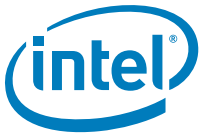
BAR	Addr	31 5	4	3	2	1	0
0	0x10	Memory CSR + FLASH BAR Low (RW - 31:17;RO - 16:4 (0x0))		0/1	1	0	0
1	0x14	Memory CSR + FLASH BAR High (RW)					
2	0x18	IO BAR (R/W - 31:5)	0	0	0	0	1/0
3	0x1C	Reserved (RO - 0)					
4	0x20	MSI-X BAR Low (RW - 31:14; RO - 13:4 (0x0))		0/1	1	0	0
5	0x24	MSI-X BAR High (RW)					

9.3.11.3 Base Address Register Fields

All base address registers have the following fields.

Table 9-6. Base Address Registers' Fields

Field	Bits	R/W	Description	
Mem / IO Space Indication	0	RO	0b = Indicates memory space. 1b = Indicates I/O. The value of bit zero of the IO BAR at offset 0x18 is loaded from the IO_Sup bit in the NVM.	
Memory Type	2:1	RO	00b = 32-bit BAR (BAR32 in the NVM equals 1b) 10b = 64-bit BAR (BAR32 in the NVM equals 0b)	
Prefetch Memory	3	RO	0b = Non-prefetchable space. 1b = Prefetchable space (device default). This bit is loaded from the PREFBAR bit in the NVM. This bit should be set only on systems that do not generate prefetchable cycles.	
Address Space (Low register for 64bit Memory BARs)	31:4	R/W	The length of the RW bits and RO 0b bits depend on the mapping window sizes. Init value of the RW fields is 0x0.	
			Mapping Window	RO bits
			Memory CSR + FLASH BAR size depends on the "Flash Size bar" and "CSR_Size" fields in the NVM.	16:4 for 128KB 17:4 for 256KB and so on...
			MSI-X space is 16KB	13:4
			I/O spaces size is 32 bytes	4



9.3.12 CardBus CIS (0x28; RO)

Not used. Hardwired to zero.

9.3.13 Subsystem Vendor ID (0x2C; RO)

This value can be loaded automatically from NVM address 0x0C at power up or reset. A value of 0x8086 is the default for this field at power up if the NVM does not respond or is not programmed.

9.3.14 Subsystem ID (0x2E; RO)

This value can be loaded automatically from NVM address 0x0B at power up with a default value of 0x0000.

9.3.15 Expansion ROM Base Address (0x30; RW)

This register is used to define the address and size information for boot-time access to the optional Flash memory. Expansion ROM is enabled by placing 0b in the *LAN Boot Disable* NVM bit. This register returns a zero value for function without an Expansion ROM window.

Field	Bit(s)	R/W	Initial Value	Description
En	0	RO	0b	1b = Enables Expansion ROM access. 0b = Disables Expansion ROM access.
Reserved	15:1	RO	0b	Always read as 0b. Writes are ignored.
Address	31:16	R/W	0b	Read-write bits are hard wired to 0b and dependent on the memory mapping window size. The LAN Expansion ROM spaces can be either 512 KB to 8 MB in powers of 2. Mapping window size is set by the *FLBAR_size* NVM field. Note: Increasing the FLBAR_size beyond 1 MB does not increase the Flash area that can be accessed through the EXPROM BAR (see Section 3.3.3.1).

9.3.16 Cap_Ptr (0x34; RO)

The *Capabilities Pointer* field (Cap_Ptr) is an 8-bit field that provides an offset in the device's PCI configuration space for the location of the first item in the Capabilities Linked List (CLL). Foxville sets this bit and implements a capabilities list to indicate that it supports PCI power management, Message Signaled Interrupts (MSIs), and PCIe extended capabilities. Its value is 0x40, which is the address of the first entry: PCI power management.

9.3.17 Interrupt Line (0x3C; RW)

Read/write register programmed by software to indicate which of the system interrupt request lines this Foxville's interrupt pin is bound to. See the PCIe definition for more details.

9.3.18 Interrupt Pin (0x3D; RO)

Hardwired to zero (indicating the use of INTA#).

9.3.19 Max_Lat/Min_Gnt (0x3E; RO)

Not used. Hardwired to zero.

9.4 PCI Capabilities

The first entry of the PCI capabilities link list is pointed by the Cap_Ptr register. The following tables describes the capabilities supported by Foxville.

Table 9-7. PCI capabilities

Address	Item	Next Pointer
0x40-47	PCI Power Management	0x50
0x50-67	Message Signaled Interrupt	0x70
0x70-8B	Extended Message Signaled Interrupt	0xA0
0xA0-DB	PCIe Capabilities	0xE0/0x00[1]
0xE0-0xE7	Vital Product Data Capability	0x00

1. Next pointer is 0x00 if the VPD area in the NVM does not exist. In mode, the PCIe capability is the last capabilities section.

9.4.1 PCI Power Management Capability

All fields are reset on full power-up. All of the fields except *PME_En* and *PME_Status* are reset on exit from D3cold state. If aux power is not supplied, the *PME_En* and *PME_Status* fields also reset on exit from D3cold state.

See the detailed description for registers loaded from the NVM at initialization time. Behavior of some fields in this section depend on the *Power Management* bit in NVM word 0x0A.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x40	Power Management Capabilities		Next Pointer (0x50)	Capability ID (0x01)
0x44	Data	Bridge Support Extensions	Power Management Control & Status	

9.4.1.1 Capability ID (0x40; RO)

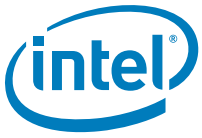
This field equals 0x01 indicating the linked list item as being the PCI Power Management registers.

9.4.1.2 Next Pointer (0x41; RO)

This field provides an offset to the next capability item in the capability list. In LAN function, a value of 0x50 points to the MSI capability.

9.4.1.3 Power Management Capabilities - PMC (0x42; RO)

This field describes Foxville's functionality at the power management states as described in the following table. Note that each device function has its own register.



Bits	Default	R/W	Description
15:11	00000b if PM is disabled in the NVM Else, 01001b or 11001b if no AUX PWR or AUX PWR is supplied respectively	RO	PME_Support - This 5-bit field indicates the power states in which the function might assert PME#. A value of 0b for any bit indicates that the function is not capable of asserting the PME# signal while in that power state. bit(11) X XXX1b - PME# can be asserted from D0 bit(14) X 1XXXb - PME# can be asserted from D3hot bit(15) 1 XXXXb - PME# can be asserted from D3cold (Value of bit 15 is a function of Aux Pwr availability and *Power Management* (PM Ena) bit in *Initialization Control Word 1* (word 0x0A) NVM word). Note: Aux Pwr is considered available if AUX_PWR pin is connected to 3.3V and *D3COLD_WAKEUP_ADVEN* NVM bit is set to 1b.
10	0b	RO	D2_Support Foxville does not support D2 state.
9	0b	RO	D1_Support Foxville does not support D1 state.
8:6	000b	RO	AUX Current – Required current defined in the Data Register.
5	1b	RO	DSI Foxville requires its device driver to be executed following transition to the D0 uninitialized state.
4	0b	RO	Reserved
3	0b	RO	PME_Clock Disabled. Hardwired to 0b.
2:0	011b	RO	Version Foxville complies with the PCI PM specification, revision 1.2.

9.4.1.4 Power Management Control / Status Register - PMCSR (0x44; R/W)

This register is used to control and monitor power management events in Foxville. Note that each device function has its own *PMCSR* register.

Bits	Default	R/W	Description
15	0b (at power up)	R/W1CS	PME_Status This bit is set to 1b when the function detects a wake-up event independent of the state of the *PME_En* bit. Writing a 1b clears this bit.
14:13	01b	RO	Data_Scale This field indicates the scaling factor to be used when interpreting the value of the Data register. This field equals 01b (indicating 0.1 watt units) if power management is enabled in the *Power Management* (PM Ena) bit in *Initialization Control Word 1* (word 0x0A) NVM word and the *Data_Select* field is set to 0, 3, 4 or 7. Otherwise, this field equals 00b.
12:9	0000b	R/W	Data_Select This four-bit field is used to select which data is to be reported through the Data register and *Data_Scale* field. These bits are writable only when power management is enabled by setting the *Power Management* (PM Ena) bit in *Initialization Control Word 1* (word 0x0A) NVM word.
8	0b (at power up)	R/WS	PME_En If power management is enabled in the NVM, writing a 1b to this register enables wake up. If power management is disabled in the NVM, writing a 1b to this bit has no effect and does not set the bit to 1b.
7:4	000000b	RO	Reserved

Bits	Default	R/W	Description
3	1b[1]	RO	No_Soft_Reset No_Soft_Reset - When set (“1”), this bit indicates that when Foxville transitions from D3hot to D0 because of modifying *Power State* bits in the *PMCSR* register, no internal reset is issued and Configuration Context is preserved. Upon transition from the D3hot to the D0 Initialized state, no additional operating system intervention is required to preserve Configuration Context beyond writing the *Power State* bits. When clear (“0”), Foxville performs an internal reset upon transitioning from D3hot to D0 via software control of the *Power State* bits in the *PMCSR* register. Configuration Context is lost when performing the soft reset. Upon transition from the D3hot to the D0 state, full re initialization sequence is needed to return the device to D0 Initialized. Regardless of this bit, devices that transition from D3hot to D0 by a system or bus segment reset returns to the device state D0 Uninitialized with only PME context preserved if PME is supported and enabled.
2	0b	RO	Reserved for PCIe.
1:0	00b	R/W	Power State 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 (cycle ignored if power management is not enabled in the NVM)

1. Loaded from NVM (See Section 6.1.1.27).

9.4.1.5 Bridge Support Extensions - PMCSR_BSE (0x46; RO)

This register is not implemented in Foxville. Values are set to 0x00.

9.4.1.6 Data Register (0x47; RO)

This optional register is used to report power consumption and heat dissipation. Reported register is controlled by the *Data_Select* field in the PMCSR and the power scale is reported in the *Data_Scale* field in the PMCSR. The data of this field is loaded from the NVM if power management is enabled in the NVM or with a default value of 0x00. The values for Foxville are read from NVM word 0x22.

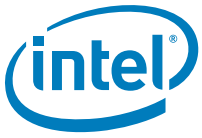
Function	D0 (Consume/Dissipate)	D3 (Consume/Dissipate)	Common
PMCSR.Data Select	0x0 / 0x4	0x3 / 0x7	0x8
Reported Value in the Data Register	LAN D0 Power field in the NVM addr 0x22	LAN D3 Power field in the NVM addr 0x22	0x0

For other *Data_Select* values, the Data register output is reserved (0x0).

9.4.2 MSI Configuration

This structure is required for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x50	Message Control (0x0180)		Next Pointer (0x70)	Capability ID (0x05)
0x54	Message Address			
0x58	Message Upper Address			



Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x5C	Reserved		Message Data	
0x60	Mask bits			
0x64	Pending bits			

9.4.2.1 Capability ID (0x50; RO)

This field equals 0x05 indicating the linked list item as being the MSI registers.

9.4.2.2 Next Pointer (0x51; RO)

This field provides an offset to the next capability item in the capability list. Its value of 0x70 points to the MSI-X capability structure.

9.4.2.3 Message Control (0x52; R/W)

The register fields are described in the following table. There is a dedicated register per PCI function to separately enable their MSI.

Bits	Default	R/W	Description
0	0b	R/W	MSI Enable If set to 1b, equals MSI. In this case, Foxville generates an MSI for interrupt assertion instead of INTx signaling.
3:1	000b	RO	Multiple Message Capable Foxville indicates a single requested message.
6:4	000b	R/W	Multiple Message Enable Foxville returns 000b by default. Programming any other value than 000b is meaningless since Foxville supports only a single MSI message.
7	1b	RO	64-bit capable A value of 1b indicates that Foxville is capable of generating 64-bit message addresses.
8	1b[1]	RO	MSI per-vector masking. A value of 1b indicates that Foxville is capable of per-vector masking. This field is loaded from the *MSI-X Configuration* (Offset 0x16) NVM word.
15:9	0b	RO	Reserved Write 0 ignore on read.

1. Default value is read from the NVM.

9.4.2.4 Message Address Low (0x54; R/W)

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.

9.4.2.5 Message Address High (0x58; R/W)

Written by the system to indicate the upper 32-bits of the address to use for the MSI memory write transaction.

9.4.2.6 Message Data (0x5C; R/W)

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.

9.4.2.7 Mask bits (0x60; R/W)

The Mask Bits and Pending Bits registers enable software to disable or defer message sending on a per-vector basis. As Foxville supports only one message, only bit 0 of these register is implemented.

Bits	Default	R/W	Description
0	0b	R/W	MSI Vector 0 Mask If set, Foxville is prohibited from sending MSI messages.
31:1	000b	RO	Reserved

9.4.2.8 Pending Bits (0x64; R/W)

Bits	Default	R/W	Description
0	0b	RO	If set, Foxville has a pending MSI message.
31:1	000b	RO	Reserved

9.4.3 MSI-X Configuration

More than one MSI-X capability structure is prohibited, but 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 a MSI-X Pending Bit Array (PBA) structure, each residing in memory space.

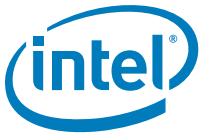
Each structure is mapped by a Base Address Register (BAR) belonging to the function, located beginning at 0x10 in configuration space. A BAR Indicator Register (BIR) indicates which BAR, and a Qword-aligned offset indicates where the structure begins relative to the base address associated with the BAR. The BAR is permitted to be either 32-bit or 64-bit, but must map to memory space. A function is permitted to map both structures with the same BAR, or to map each structure with a different BAR.

The MSI-X table structure, listed in Section 9.4.3.6, typically contains multiple entries, each consisting of several fields: message address, message upper address, message data, and vector control. Each entry is capable of specifying a unique vector.

The PBA structure, described in the same section, contains the function's pending bits, one per Table entry, organized as a packed array of bits within Qwords. Note that the last Qword might not be fully populated.

To request service using a given MSI-X table entry, a function performs a Dword memory write transaction using:

- The contents of the Message Data field entry for data.
- The contents of the Message Upper Address field for the upper 32 bits of the address.
- The contents of the Message Address field entry for the lower 32 bits of the address.



A memory read transaction from the address targeted by the MSI-X message produces undefined results.

The MSI-X table and MSI-X PBA are permitted to co-reside within a naturally aligned 4 KB address range, though they must not overlap with each other.

MSI-X table entries and Pending bits are each numbered 0 through N-1, where N-1 is indicated by the Table Size field in the MSI-X Message Control register. For a given arbitrary MSI-X table entry K, its starting address can be calculated with the formula:

Entry starting address = Table base + K*16

For the associated Pending bit K, its address for Qword access and bit number within that Qword can be calculated with the formulas:

Qword address = PBA base + (K div 64)*8
Qword bit# = K mod 64

Software that chooses to read Pending bit K with Dword accesses can use these formulas:

Dword address = PBA base + (K div 32)*4
Dword bit# = K mod 32

Foxville also supports the table-less MSI-X mode, where a single interrupt vector is provided. The MSI-X table and MSI-X PBA are not used. Instead, the capability structure includes several additional fields (Message Address, Message Address Upper, and Message Data) for vector configuration. Foxville embeds the number of the original MSI-X vectors (i.e. the vectors supported if the number of vectors was not limited to 1) in the LSB bits of the Message Data field.

Table 9-8. MSI-X Capability Structure

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x70	Message Control (0x0004)		Next Pointer (0xA0)	Capability ID (0x11)
0x74	Table Offset			
0x78	PBA offset			

9.4.3.1 Capability ID (0x70; RO)

This field equals 0x11 indicating the linked list item as being the MSI-X registers.

9.4.3.2 Next Pointer (0x71; RO)

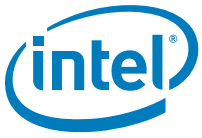
This field provides an offset to the next capability item in the capability list. Its value of 0xA0 points to the PCIe capability.

9.4.3.3 Message Control (0x72; R/W)

The register fields are described in the following table. There is a dedicated register per PCI function to separately configure their MSI-X functionality.

Bits	Default	R/W	Description
10:0	0x004[1]	RO	TS - Table Size System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. For example, a returned value of 0x00F indicates a table size of 16. Foxville supports 5 MSI-X vectors. This field is loaded from the *MSI-X Configuration* (Offset 0x16) NVM word.
13:11	000b	RO	Reserved Always return 000b on read. Write operation has no effect.
14	0b	R/W	FM - Function Mask If set to 1b, all of the vectors associated with the function are masked, regardless of their per-vector *Mask* bit states. If set to 0b, each vector's *Mask* bit determines whether the vector is masked or not. Setting or clearing the *MSI-X Function Mask* bit has no effect on the state of the per-vector *Mask* bits.
15	0b	R/W	En - MSI-X Enable If set to 1b and the *MSI Enable* bit in the MSI Message Control (MMC) 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 software device driver is prohibited from writing this bit to mask a function's service request. If set to 0b, the function is prohibited from using MSI-X to request service.

1. Default value is read from the NVM.



9.4.3.4 MSI-X Table Offset (0x74; R/W)

Bits	Default	Type	Description
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 table. The lower three table BIR bits are masked off (set to zero) by software to form a 32-bit Qword-aligned offset.
2:0	0x3/0x4	RO	Table 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 table into memory space.When BIR values: 0...5 correspond to BARs 0x10...0x 24 respectively. operating in 32bit MMIO (BAR32 = 1) a BIR value of 3 indicates that the table is mapped in BAR 3 (address 0x1C). When operating in 64bit MMIO (BAR32 = 0) a BIR value of 4 indicates that the table is mapped in BAR 4 (address 0x20).

9.4.3.5 MSI-X Pending Bit Array - PBA Offset (0x78; R/W)

Bits	Default	Type	Description
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.
2:0	0x3	RO	PBA BIR: Indicates which one of a function's Base Address registers, located beginning at 10h in Configuration Space, is used to map the function's MSI-X PBA into Memory Space. The BIR of the PBA is identical to the BIR of the MSI-X table.

9.4.3.6 MSI-X Interrupt Vector and Pending Bit Array Tables

The interrupt vector table consist of a list of the address and data of the interrupt vectors. It is shown in the Table 9-9 below. The pending bit array indicates which vectors has a pending interrupts. The structure is shown in Table 9-10.

Table 9-9. MSI-X Table Structure

DWORD3 MSIXTVCTRL	DWORD2 MSIXTMSG	DWORD1 MSIXTUADD	DWORD0 MSIXTADD	Entry Number	BAR 3 - Offset
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 0	Base = 0x0000
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 1	Base + 1*16
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 2	Base + 2*16
...	...	...	...	...	...
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry (4)	Base + 4 *16

Table 9-10. MSI-X PBA Structure

MSIXPBA[63:0]	Qword Number	BAR 3 - Offset
Pending Bits 0 through 63 (only bits 0 through bit 4 are meaningful)	QWORD0	Base = 0x2000

9.4.3.6.1 MSI-X Table Entry Lower Address - MSIXTADD (BAR3: 0x0000 + 0x10*n [n=0...4]; R/W)

Register is reset by LAN_PWR_GOOD, PCIe Reset and FLR only.

Field	Bit(s)	Initial Value	Description
Message Address LSB (RO)	1:0	0x0	For proper Dword alignment, software must always write 0b,Äôs to these two bits. Otherwise, the result is undefined.
Message Address	31:2	0x0	System-Specific 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 for the memory write transaction.

9.4.3.6.2 MSI-X Table Entry Upper Address - MSIXTUADD (BAR3: 0x0004 + 0x10*n [n=0...4]; R/W)

Register is reset by LAN_PWR_GOOD, PCIe Reset and FLR only.

Field	Bit(s)	Initial Value	Description
Message Address	31:0	0x0	System-Specific Message Upper Address.

9.4.3.6.3 MSI-X Table Entry Message - MSIXTMSG (BAR3: 0x0008 + 0x10*n [n=0...4]; R/W)

Register is reset by LAN_PWR_GOOD, PCIe Reset and FLR only.

Field	Bit(s)	Initial Value	Description
Message Data	31:0	0x0	System-Specific Message Data. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the data written during the memory write transaction. In contrast to message data used for MSI messages, the low-order message data bits in MSI-X messages are not modified by the function.

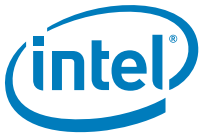
9.4.3.6.4 MSI-X Table Entry Vector Control - MSIXTVCTRL (BAR3: 0x000C + 0x10*n [n=0...4]; R/W)

Register is reset by LAN_PWR_GOOD, PCIe Reset and FLR only.

Field	Bit(s)	Initial Value	Description
Mask	0	1b	When this bit is set, the function is prohibited from sending a message using this MSI-X table entry. However, any other MSI-X table entries programmed with the same vector are still capable of sending an equivalent message unless they are also masked.
Reserved	31:1	0x0	Reserved. Write 0x0, ignore on read.

9.4.3.6.5 MSIXPBA Bit Description – MSIXPBA (BAR3: 0x2000; RO)

Register is reset by LAN_PWR_GOOD, PCIe Reset and FLR only.



Field	Bit(s)	Initial Value	Description
Pending Bits	4:0	0x0	For each pending bit that is set, the function has a pending message for the associated MSI-X table entry. Pending bits that have no associated MSI-X table entry are reserved.
Reserved	31:5	0x0	Reserved. Write 0x0, ignore on read.

Bits	Default	Type	Description
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 table. The lower three table BIR bits are masked off (set to zero) by software to form a 32-bit Qword-aligned offset.
2:0	0x3/0x4	RO	Table 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 table into memory space.When BIR values: 0...5 correspond to BARs 0x10...0x 24 respectively. operating in 32bit MMIO (BAR32 = 1) a BIR value of 3 indicates that the table is mapped in BAR 3 (address 0x1C). When operating in 64bit MMIO (BAR32 = 0) a BIR value of 4 indicates that the table is mapped in BAR 4 (address 0x20).

9.4.4 Vital Product Data Registers

Foxville supports access to a VPD structure stored in the NVM using the following set of registers.

Note:

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0xE0	VPD address		Next Pointer (0x00)	Capability ID (0x03)
0xE4	VPD data			

9.4.4.1 Capability ID (0xE0; RO)

This field equals 0x3 indicating the linked list item as being the VPD registers.

9.4.4.2 Next Pointer (0xE1; RO)

Offset to the next capability item in the capability list. A 0x00 value indicates that it is the last item in the capability-linked list.

9.4.4.3 VPD Address (0xE2; RW)

Dword-aligned byte address of the VPD area in the NVM to be accessed. The register is read/write with the initial value at power-up indeterminate.

Bits	Default	R/W	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 with the initial value at power-up indeterminate. The two LSBs are RO as zero. This is the address relative to the start of the VPD area. As the maximal size supported by Foxville is 1024 bytes, bits 14:10 should always be 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 the device when data is valid. 1b = Write. Cleared by the device when data is written to the NVM. The VPD address and data should not be modified before the action completes.

9.4.4.4 VPD Data (0xE4; RW)

This register contains the VPD read/write data.

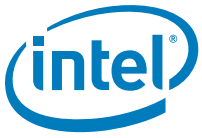
Bits	Default	R/W	Description
31:0	X	RW	VPD Data VPD data can be read or written through this register. The LSB of this register (at offset four in this capability structure) corresponds to the byte of VPD at the address specified by the VPD Address register. The data read from or written to this register 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.

9.4.5 PCIe Configuration Registers

PCIe provides two mechanisms to support native features:

- PCIe defines a PCI capability pointer indicating support for PCIe.
- PCIe extends the configuration space beyond the 256 bytes available for PCI to 4096 bytes.

Foxville implements the PCIe capability structure for endpoint devices as follows:



9.4.5.1 Capability ID (0xA0; RO)

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0xA0	PCI Express Capability Register (0x0002)		Next Pointer (0xE0/ 0x00)	Capability ID (0x10)
0xA4	Device Capability			
0xA8	Device Status		Device Control	
0xAC	Link Capabilities			
0xB0	Link Status		Link Control	
0xB4	Reserved			
0xB8	Reserved		Reserved	
0xBC	Reserved			
0xC0	Reserved		Reserved	
0xC4	Device Capabilities 2			
0xC8	Reserved		Device Control 2	
0xCC	Reserved			
0xD0	Link Status 2		Link Control 2	
0xD4	Reserved			
0xD8	Reserved		Reserved	

This field equals 0x10 indicating the linked list item as being the PCIe Capabilities registers.

9.4.5.2 Next Pointer (0xA1; RO)

Offset to the next capability item in the capability list. Its value of 0xE0 points to the VPD structure. If VPD is disabled, or operating in mode, a value of 0x00 value indicates that it is the last item in the capability-linked list.

9.4.5.3 PCIe CAP (0xA2; RO)

The PCIe capabilities register identifies the PCIe device type and associated capabilities. This is a read only register.

Bits	Default	R/W	Description
3:0	0010b	RO	Capability Version Indicates the PCIe capability structure version number. Foxville supports both version 1 and version 2 as loaded from the PCIe *Capability Version* bit in the NVM.
7:4	0000b	RO	Device/Port Type Indicates the type of PCIe function. a native PCI function with a value of 0000b.
8	0b	RO	Slot Implemented Foxville does not implement slot options therefore this field is hardwired to 0b.
13:9	00000b	RO	Interrupt Message Number Foxville does not implement multiple MSI interrupts, therefore this field is hardwired to 0x0.
15:14	00b	RO	Reserved

9.4.5.4 Device Capabilities (0xA4; RO)

This register identifies the PCIe device specific capabilities. It is a read only register.

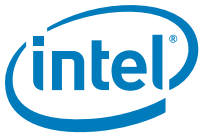
Bits	R/W	Default	Description
2:0	RO	010b	Max Payload Size Supported This field indicates the maximum payload that Foxville can support for TLPs. It is loaded from the NVM's *PCIe Init Configuration 3* word, 0x1A (with a default value of 512 bytes. See Section 6.1.1.26).
4:3	RO	00b	Phantom Function Supported Not supported by Foxville.
5	RO	0b	Extended Tag Field Supported Max supported size of the *Tag* field. Foxville supported 5-bit *Tag* field.
8:6	RO	011b	Endpoint L0s Acceptable Latency This field indicates the acceptable latency that Foxville can withstand due to the transition from the L0s state to the L0 state. value loaded from the NVM *PCIe Init Configuration 1* word, 0x18 (See Section 6.1.1.24).
11:9	RO	110b	Endpoint L1 Acceptable Latency This field indicates the acceptable latency that Foxville can withstand due to the transition from the L1 state to the L0 state. value loaded from the NVM *PCIe L1 Exit latencies* word, 0x14 (See Section 6.1.1.21).
12	RO	0b	Attention Button Present Hardwired in Foxville to 0b.
13	RO	0b	Attention Indicator Present Hardwired in Foxville to 0b.
14	RO	0b	Power Indicator Present Hardwired in Foxville to 0b.
15	RO	1b	Role-Based Error Reporting This bit, when set, indicates that Foxville implements the functionality originally defined in the Error Reporting ECN for PCIe Base Specification 1.0a and later incorporated into PCIe Base Specification 1.1. Set to 1b in Foxville.
17:16	RO	000b	Reserved
25:18	RO	0x00	Slot Power Limit Value Hardwired in Foxville to 0x00, as Foxville consumes less than the 25 W allowed for its form factor.
27:26	RO	00b	Slot Power Limit Scale Hardwired in Foxville to 0b, as Foxville consumes less than the 25 W allowed for its form factor.
28	RO	1b[1]	Function Level Reset (FLR) Capability A value of 1b indicates the function supports the optional FLR mechanism.
31:29	RO	000b	Reserved

1. Loaded from NVM.

9.4.5.5 Device Control (0xA8; RW)

This register controls the PCIe specific parameters.

Bits	R/W	Default	Description
0	RW	0b	Correctable Error Reporting Enable Enable report of correctable errors.
1	RW	0b	Non-Fatal Error Reporting Enable Enable report of non fatal errors.
2	RW	0b	Fatal Error Reporting Enable Enable report of fatal errors.



Bits	R/W	Default	Description
3	RW	0b	Unsupported Request Reporting Enable Enable report of unsupported requests error.
4	RW	1b	Enable Relaxed Ordering If this bit is set, Foxville is permitted to set the *Relaxed Ordering* bit in the attribute field of write transactions that do not need strong ordering. For more details, refer to the description about the RO_DIS bit in the CTRL_EXT register bit in Section 8.2.3.
7:5	RW	000b (128 bytes)	Max Payload Size This field sets maximum TLP payload size for Foxville. As a receiver, Foxville must handle TLPs as large as the set value. As a transmitter, Foxville must not generate TLPs exceeding the set value. The max payload size supported in Foxville Device capabilities register indicates permissible values that can be programmed. ***Note:*** According to PCIe spec, this field shall not be reset on FLR.
8	R0	0b	Extended Tag field Enable Not implemented in Foxville.
9	R0	0b	Phantom Functions Enable Not implemented in Foxville.
10	RWS	0b	Auxiliary Power PM Enable When set, enables Foxville to draw AUX power independent of PME AUX power.
11	RW	1b	Enable No Snoop Snoop is gated by *NONSNOOP* bits in the GCR register in the CSR space.
14:12	RW	010b	Max Read Request Size - this field sets maximum read request size for the Device as a requester. 000b = 128 bytes 001b = 256 bytes. 010b = 512 bytes (the default value). 011b = 1 KB. 100b = Reserved. 101b = Reserved. 110b = Reserved. 111b = Reserved.
15	RW	0b	Initiate Function Level Reset A write of 1b initiates an FLR to the function. The value read by software from this bit is always 0b.

9.4.5.6 Device Status (0xAA; R/W1C)

This register provides information about PCIe device's specific parameters.

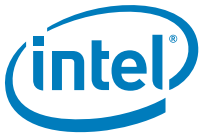
Bits	R/W	Default	Description
0	R/W1C	0b	Correctable Error Detected Indicates status of correctable error detection.
1	R/W1C	0b	Non-Fatal Error Detected Indicates status of non-fatal error detection.
2	R/W1C	0b	Fatal Error Detected Indicates status of fatal error detection.
3	R/W1C	0b	Unsupported Request Detected Indicates that Foxville received an unsupported request.
4	RO	0b	Aux Power Detected If aux power is detected, this field is set to 1b. It is a straping signal from the periphery.

Bits	R/W	Default	Description
5	RO	0b	Transactions Pending Indicates whether Foxville has any transaction pending. (Transactions include completions for any outstanding non-posted request for all used traffic classes).
15:6	RO	0x00	Reserved

9.4.5.7 Link Capabilities Register (0xAC; RO)

This register identifies PCIe link specific capabilities. This is a read only register

Bits	Rd/Wr	Default	Description
3:0	RO	0010b	Max Link Speed This field indicates the supported Link speed(s) of the associated link port. Defined encodings are: 0001b = 2.5 Gb/s Link speed supported. 0010b = Not supported (5 Gb/s and 2.5 Gb/s Link speeds)
9:4	RO	0x01	Max Link Width Indicates the maximum link width. Foxville can support by 1 link width. Relevant encoding: 000000b = Reserved. 000001b = x1. 000010b = x2 Not supported. 000100b = x4 Not supported.
11:10	RO	11b	Active State Power Management (ASPM) Support – This field indicates the level of ASPM supported on Foxville PCI Express Link. Defined encodings are: 00b = No ASPM Support. 01b = L0s Supported. 10b = L1 Supported. <-- default value in the NVM (*Act_Stat_PM_Sup*) 11b = L0s and L1 Supported. This field is loaded from field *Act_Stat_PM_Sup* in the NVM *PCIe Init Configuration 3* word, 0x1A. (See Section 6.1.1.26).
14:12	RO	Usage depended. See default values in Section 6.1.1.24.	L0s Exit Latency Indicates the exit latency from L0s to L0 state. 000b = Less than 64ns. 001b = 64ns – 128ns. 010b = 128ns – 256ns. 011b = 256ns - 512ns. 100b = 512ns - 1 µs. 101b = 1 µs – 2 µs. 110b = 2 µs – 4 µs. 111b = Reserved. Depending on usage of common clock or separate clock the value of this field is loaded from PCIe Init Config 1 NVM word, 0x18 (See Section 6.1.1.24).
17:15	RO	Usage depended. See default values in Section 6.1.1.21.	L1 Exit Latency Indicates the exit latency from L1 to L0 state. 000b = Less than 1 µs. 001b = 1 µs - 2 µs. 010b = 2 µs - 4 µs. 011b = 4 µs - 8 µs. 100b = 8 µs - 16 µs. 101b = 16 µs - 32 µs. 110b = 32 µs - 64 µs. 111b = L1 transition not supported. Depending on usage of common clock or separate clock the value of this field is loaded from *PCIe L1 Exit latencies* NVM word, 0x14 (See Section 6.1.1.21).



Bits	Rd/Wr	Default	Description
18	RO	0b	Clock Power Management Status Not supported in Foxville. RO as zero.
19	RO	0b	Surprise Down Error Reporting Capable Status Not supported in Foxville. RO as zero
20	RO	0b	Data Link Layer Link Active Reporting Capable Status Not supported in Foxville. RO as zero.
21	RO	0b	Link Bandwidth Notification Capability Status Not supported in Foxville. RO as zero.
22	RO	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	00b	Reserved
31:24	HwInit	0x0	Port Number The PCIe port number for the given PCIe link. Field is set in the link training phase.

9.4.5.8 Link Control Register (0xB0; RO)

This register controls PCIe link specific parameters.

Bits	R/W	Default	Description
1:0	RW	00b	Active State Power Management (ASPM) Control – This field controls the level of Active State Power Management (ASPM) supported on Foxville PCI Express Link. Defined encodings are: 00b = PM disabled. 01b = L0s entry supported. 10b = L1 Entry Enabled. 11b = L0s and L1 supported. ***Note:*** "L0s Entry Enabled" enables the Transmitter to enter L0s is supported. If L0s is supported, the Receiver must be capable of entering L0s even when the Transmitter is disabled from entering L0s (00b or 10b). According to PCIe spec, this field shall not be reset on FLR.
2	RO	0b	Reserved
3	RW	0b	Read Completion Boundary Read Completion Boundary (RCB) – Optionally Set by configuration software to indicate the RCB value of the Root Port Upstream from the Endpoint or Bridge. Defined encodings are: 0b = 64 byte 1b = 128 byte Configuration software must only Set this bit if the Root Port Upstream from the Endpoint or Bridge reports an RCB value of 128 bytes (a value of 1b in the Read Completion Boundary bit).
4	RO	0b	Link Disable Not applicable for endpoint devices; hardwired to 0b.
5	RO	0b	Retrain Clock Not applicable for endpoint devices; hardwired to 0b.
6	RW	0b	Common Clock Configuration When this bit is set, it indicates that Foxville and the component at the other end of the link are operating with a common reference clock. A value of 0b indicates that both operate with an asynchronous clock. This parameter affects the L0s exit latencies. ***Note:*** According to PCIe spec, this field shall not be reset on FLR.

Bits	R/W	Default	Description
7	RW	0b	Extended Synch When this bit is set, it forces an extended Tx of a FTS ordered set in FTS and an extra TS1 at exit from L0s prior to enter L0. ***Note:*** According to PCIe spec, this field shall not be reset on FLR.
8	RO	0b	Enable Clock Power Management Not supported in Foxville. RO as zero.
9	RO	0b	Hardware Autonomous Width Disable Not supported in Foxville. RO as zero.
10	RO	0b	Link Bandwidth Management Interrupt Enable Not supported in Foxville. RO as zero.
11	RO	0b	Link Autonomous Bandwidth Interrupt Enable Not supported in Foxville. RO as zero.
15:12	RO	0000b	Reserved

9.4.5.9 Link Status (0xB2; RO)

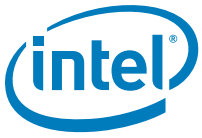
This register provides information about PCIe link specific parameters. This is a read only register.

Bits	R/W	Default	Description
3:0	RO	0010b	Link Speed This field indicates the negotiated link speed of the given PCIe link. Defined encodings are: 0001b = 2.5 Gb/s PCIe link. 0010b = 5 Gb/s PCIe link. All other encodings are reserved.
9:4	RO	000001b	Negotiated Link Width Indicates the negotiated width of the link. Relevant encoding for Foxville are: 000001b = x1 000010b = Not supported (x2) 000100b = Not supported (x4)
10	RO	0b	Reserved (was: Link Training Error)
11	RO	0b	Link Training Indicates that link training is in progress.
12	HwInit	1b	Slot Clock Configuration When set, indicates that Foxville uses the physical reference clock that the platform provides on the connector. This bit must be cleared if Foxville uses an independent clock. The Slot Clock Configuration bit is loaded from the *Slot_Clock_Cfg* bit in *PCIe Init Configuration 3* Word (Word 0x1A) NVM word.
13	RO	0b	Data Link Layer Link Active Not supported in Foxville. RO as zero.
14	RO	0b	Link Bandwidth Management Status Not supported in Foxville. RO as zero.
15	RO	0b	Reserved

9.4.5.10 Reserved (0xB4-0xC0; RO)

Unimplemented reserved registers not relevant to PCIe endpoint.

The following registers are supported only if the capability version is two and above.



9.4.5.11 Device Capabilities 2 (0xC4; RO)

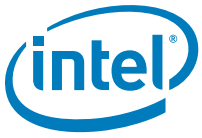
This register identifies PCIe device specific capabilities.
.

Bit Location	R/W	Default	Description
3:0	RO	1111b	Completion Timeout Ranges Supported This field indicates Foxville support for the optional completion timeout programmability mechanism. This mechanism enables system software to modify the completion timeout value. Description of the mechanism can be found in Section 3.1.3.2. Four time value ranges are defined: • Range A = 50 µs to 10 ms • Range B = 10 ms to 250 ms • Range C = 250 ms to 4 s • Range D = 4 s to 64 s A value of 1111b indicates Foxville supports ranges A, B, C, & D.
4	RO	1b	Completion Timeout Disable Supported A value of 1b indicates support for the completion timeout disable mechanism.
5	RO	0b	ARI Forwarding Supported Applicable only to switch downstream ports and root ports; must be set to 0b for other function types.
6	RO	0b	AtomicOp Routing Supported - not supported in Foxville.
7	RO	0b	32-bit AtomicOp Completer Supported – not supported in Foxville.
8	RO	0b	64-bit AtomicOp Completer Supported – not supported in Foxville.
9	RO	0b	128-bit CAS Completer Supported – not supported in Foxville.
10	RO	0b	No RO-enabled PR-PR Passing – not supported in Foxville.
11	RO	1b	LTR Mechanism Supported – A value of 1b indicates support for the optional Latency Tolerance Requirement Reporting (LTR) mechanism capability. Note: Value loaded from LTR_EN bit in Initialization Control Word 1 NVM word.
13:12	RO	00b	TPH Completer supported - Foxville does not use the hints as a completer
17:14	RO	0x0	Reserved
19:18	RO	10b	00b - OBFF Not Supported
31:20	RO	0x0	Reserved

9.4.5.12 Device Control 2 (0xC8; RW)

This register controls PCIe specific parameters.

Bit location	R/W	Default	Description
3:0	RW	0000b	Completion Timeout Value[1] In devices that support completion timeout programmability, this field enables system software to modify the completion timeout value. Encoding: • 0000b = Allowable default range: 50 µs to 50 ms. It is strongly recommended that the completion timeout mechanism not expire in less than 10 ms. Actual completion timeout range supported in Foxville is 16 ms to 32 ms. Values available if Range A (50 µs to 10 ms) programmability range is supported: • 0001b = Allowable range is 50 µs to 100 µs. Actual completion timeout range supported in Foxville is 50 µs to 100 µs. • 0010b = Allowable range is 1 ms to 10 ms. Actual completion timeout range supported in Foxville is 1 ms to 2 ms. Values available if Range B (10 ms to 250 ms) programmability range is supported: • 0101b = Allowable range is 16 ms to 55 ms. Actual completion timeout range supported in Foxville is 16 ms to 32 ms. • 0110b = Allowable range is 65 ms to 210 ms. Actual completion timeout range supported in Foxville is 65 ms to 130 ms. Values available if Range C (250 ms to 4 s) programmability range is supported: • 1001b = Allowable range is 260 ms to 900 ms. Actual completion timeout range supported in Foxville is 260 ms to 520 ms. • 1010b = Allowable range is 1 s to 3.5 s. Actual completion timeout range supported in Foxville is 1 s to 2 s. Values available if the Range D (4 s to 64 s) programmability range is supported: • 1101b = Allowable range is 4 s to 13 s. Actual completion timeout range supported in Foxville is 4 s to 8 s. • 1110b = Allowable range is 17 s to 64 s. Actual completion timeout range supported in Foxville is 17 s to 34 s. Values not defined are reserved. Software is permitted to change the value in 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 when this value was changed or when each request was issued. The default value for this field is 0000b.
4	RW	0b	Completion Timeout Disable When set to 1b, this bit disables the completion timeout mechanism. Software is permitted to set or clear this bit at any time. When set, the completion timeout detection mechanism is disabled. If there are outstanding requests when the bit is cleared, it is permitted but not required for hardware to apply the completion timeout mechanism to the outstanding requests. If this is done, it is permitted to base the start time for each request on either the time this bit was cleared or the time each request was issued. The default value for this bit is 0b.
5	RO	0b	Alternative RID Interpretation (ARI) Forwarding Enable Applicable only to switch devices.
6	RO	0b	AtomicOp Requester Enable - not supported in Foxville.
7	RO	0b	AtomicOp Egress Blocking - not supported in Foxville.
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 Completion it initiates
10	RW	0b	LTR Mechanism Enable – When Set to 1b, this bit enables the Latency Tolerance Requirement Reporting (LTR) mechanism. Notes: • If Value of *LTR_EN* bit in *Initialization Control Word 1* NVM word is 0, then bit is RO with a value of 0b.



Bit location	R/W	Default	Description
12:11	RO	0x0	Reserved.
14:13	RO	00b	00b - OBFF Disabled
15	RO	0	Reserved.

1. The completion timeout value must be programmed correctly in PCIe configuration space (in Device Control 2 Register); the value must be set above the expected maximum latency for completions in the system in which Foxville is installed. This ensures that Foxville receives the completions for the requests it sends out, avoiding a completion timeout scenario. It is expected that the system BIOS sets this value appropriately for the system.

9.4.5.13 Link Control 2 (0xD0; RW)

Bits	R/W	Default	Description
3:0	RWS	0010b	Target Link Speed. This field is used to set the target compliance mode speed when software is using the *Enter Compliance* bit to force a link into compliance mode. Defined encodings are: 0001b = 2.5 Gb/s Target Link Speed. 0010b = 5 Gb/s Target Link Speed. All other encodings are reserved. If a value is written to this field that does not correspond to a speed included in the *Max Link Speed* field, the result is undefined. The default value of this field is the highest link speed supported by Foxville (as reported in the *Max Link Speed* field of the Link Capabilities register).
4	RWS	0b	Enter Compliance. Software is permitted to force a link to enter compliance mode at the speed indicated in the *Target Link Speed* field by setting this bit to 1b in both components on a link and then initiating a hot reset on the link. The default value of this field following a fundamental reset is 0b.
5	RWS	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.
9:7	RWS	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 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 n = 111b reserved. ***Note:*** No support to half-swing (low-swing).
10	RWS	0b	Enter Modified Compliance When this bit is set to 1b, the device transmits modified compliance pattern if the LTSSM enters Polling.Compliance state.

Bits	R/W	Default	Description
11	RWS	0b	Compliance SOS When set to 1b, the LTSSM is required to send SOS periodically in between the (modified) compliance patterns.
15:12	RWS	0b	Compliance De-emphasis This bit sets the de-emphasis level in Polling.Compliance state if the entry occurred due to the Enter Compliance bit being 1b. Possible Encoding Valuess: 0001b -3.5 dB 0000b -6 dB When the Link is operating at 2.5 GT/s, the setting of this bit has no effect.

9.4.5.14 Link Status 2 (0xD2; RO)

Bits	R/W	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 operating at 2.5 GT/s speed Encodings: 1b -3.5 dB 0b -6 dB
15:1	RO	0x0	Reserved

9.5 PCIe Extended Configuration Space

PCIe extended 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. Foxville decodes an additional 4-bits (bits 27:24) to provide the additional configuration space as shown in Table 9-11. PCIe reserves the remaining 4 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.

Table 9-11. PCIe Extended Configuration Space

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.

Foxville supports the following PCIe extended capabilities.

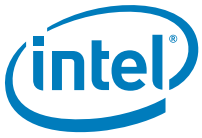


Table 9-12. PCIe Extended Capability Structure

Capability	Offset	Next Header[1]
Advanced Error Reporting	0x100	0x140
Serial Number	0x140	0x1C0
Latency Tolerance Requirement Reporting	0x1C0	0x1F0
L1 Sub states	0x1E0	0x000
Precision Time Measurement	0x1F0	0x1E0

1. Some of the capabilities might be skipped if disabled via NVM.

9.5.1 Advanced Error Reporting (AER) Capability

The PCIe AER capability is an optional extended capability to support advanced error reporting. The following table lists the PCIe AER extended capability structure for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x100	Next Capability Ptr. (0x140)[1]	Version (0x2)	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			

1. This value might change if the SEID capability is disabled. In this case the next header is the next enabled feature.

9.5.1.1 PCIe CAP ID (0x100; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x0001	Extended Capability ID PCIe extended capability ID indicating AER capability.
19:16	RO	0x2[1]	AER Capability Version PCIe AER extended capability version number.
31:20	RO	0x140	Next Capability Pointer Next PCIe extended capability pointer. A value of 0x140 points to the serial ID capability.

1. Loaded from NVM (See Section 6.1.1.30).

9.5.1.2 Uncorrectable Error Status (0x104; R/W1CS)

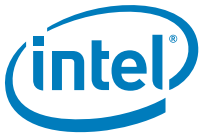
The Uncorrectable Error Status register reports error status of individual uncorrectable error sources on a PCIe device. An individual error status bit that is set to 1b indicates that a particular error occurred; software can clear an error status by writing a 1b to the respective bit.

Bit Location	Attribute	Default Value	Description
3:0	RO	0x0	Reserved
4	R/W1CS	0b	Data Link Protocol Error Status
5	RO	0b	Surprise Down Error Status (Optional) Not supported in Foxville.
11:6	RO	0x0	Reserved
12	R/W1CS	0b	Poisoned TLP Status
13	R/W1CS	0b	Flow Control Protocol Error Status
14	R/W1CS	0b	Completion Timeout Status
15	R/W1CS	0b	Completer Abort Status
16	R/W1CS	0b	Unexpected Completion Status
17	R/W1CS	0b	Receiver Overflow Status
18	R/W1CS	0b	Malformed TLP Status
19	R/W1CS	0b	ECRC Error Status
20	R/W1CS	0b	Unsupported Request Error Status
21	RO	0b	ACS Violation Status Not supported in Foxville.
22	RO	0b	Uncorrectable Internal Error Status (Optional) Not supported in Foxville.
23	RO	0b	MC Blocked TLP Status (Optional) Not supported in Foxville.
24	RO	0b	AtomicOps Egress Blocked Status (Optional) Not supported in Foxville.
25	RO	0b	TLP Prefix Blocked Error Status (Optional) Not supported in Foxville.
31:26	RO	0x0	Reserved

9.5.1.3 Uncorrectable Error Mask (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. There is a mask bit per bit in the Uncorrectable Error Status register.

Bit Location	Attribute	Default Value	Description
3:0	RO	0x0	Reserved
4	RWS	0b	Data Link Protocol Error Mask
5	RO	0b	Surprise Down Error Mask (Optional) Not supported in Foxville.
11:6	RO	0x0	Reserved
12	RWS	0b	Poisoned TLP 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



Bit Location	Attribute	Default Value	Description
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
21	RO	0b	ACS Violation Mask Not supported in Foxville.
22	RO	0b	Uncorrectable Internal Error Mask (Optional) Not supported in Foxville.
23	RO	0b	MC Blocked TLP Mask (Optional) Not supported in Foxville.
24	RO	0b	AtomicOps Egress Blocked Mask (Optional) Not supported in Foxville.
25	RO	0b	TLP Prefix Blocked Error Mask (Optional) Not supported in Foxville.
31:26	RO	0x0	Reserved

9.5.1.4 Uncorrectable Error Severity (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
3:0	RO	0001b	Reserved
4	RWS	1b	Data Link Protocol Error Severity
5	RO	1b	Surprise Down Error Severity (Optional) Not supported in Foxville.
11:6	RO	0x0	Reserved
12	RWS	0b	Poisoned TLP 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 in Foxville.
22	RO	1b	Uncorrectable Internal Error Severity (Optional) Not supported in Foxville.
23	RO	0b	MC Blocked TLP Severity (Optional) Not supported in Foxville.

Bit Location	Attribute	Default Value	Description
24	RO	0b	AtomicOps Egress Blocked Severity (Optional) Not supported in Foxville.
25	RO	0b	TLP Prefix Blocked Error Severity (Optional) Not supported in Foxville.
31:26	RO	0x0	Reserved

9.5.1.5 Correctable Error Status (0x110; R/W1CS)

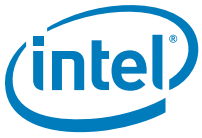
The Correctable Error Status register reports error status of individual correctable error sources on a PCIe device. When an individual error status bit is set to 1b, it indicates that a particular error occurred; software can clear an error status by writing a 1b to the respective bit.

Bit Location	Attribute	Default Value	Description
0	R/W1CS	0b	Receiver Error Status
5:1	RO	0x0	Reserved
6	R/W1CS	0b	Bad TLP Status
7	R/W1CS	0b	Bad DLLP Status
8	R/W1CS	0b	REPLAY_NUM Rollover Status
11:9	RO	000	Reserved
12	R/W1CS	0b	Replay Timer Timeout Status
13	R/W1CS	0b	Advisory Non-Fatal Error Status
14	RO	0b	Corrected Internal Error Status (Optional) Not supported in Foxville.
15	RO	0b	Header Log Overflow Status (Optional) Not supported in Foxville.
31:16	RO	0x0	Reserved

9.5.1.6 Correctable Error Mask (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	RO	0x0	Reserved
6	RWS	0b	Bad TLP Mask
7	RWS	0b	Bad DLLP Mask
8	RWS	0b	REPLAY_NUM Rollover Mask
11:9	RO	000b	Reserved
12	RWS	0b	Replay Timer Timeout Mask
13	RWS	1b	Advisory Non-Fatal Error Mask. This bit is Set by default to enable compatibility with software that does not comprehend Role-Based Error Reporting.



Bit Location	Attribute	Default Value	Description
14	RO	0b	Corrected Internal Error Mask (Optional) Not supported in Foxville.
15	RO	0b	Header Log Overflow Mask (Optional) Not supported in Foxville.
31:16	RO	0x0	Reserved

9.5.1.7 Advanced Error Capabilities and Control Register (0x118; RWS)

Bit Location	Attribute	Default Value	Description
4:0	ROS	0x0	First Error Pointer The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register.
5	RO	1b	ECRC Generation Capable This bit indicates that Foxville is capable of generating ECRC. This bit is loaded from NVM PCIe Control 2 word (Word 0x28).
6	RWS	0b	ECRC Generation Enable When set, enables ECRC generation.
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 PCIe Control 2 word (Word 0x28).
8	RWS	0b	ECRC Check Enable When set, enables ECRC checking.
9	RO	0b	Multiple Header Recording Capable – If Set, this bit indicates that the Function is capable of recording more than one error header.
10	RO	0b	This bit enables the Function to record more than one error header.
11	RO	0b	TLP Prefix Log Present If Set and the First Error Pointer is valid, indicates that the TLP Prefix Log register contains valid information. If Clear or if First Error Pointer is invalid, the TLP Prefix Log register is undefined. Default value of this bit is 0b. This bit is RsvdP if the End-End TLP Prefix Supported bit is Clear.
31:12	RO	0x0	Reserved

9.5.1.8 Header Log (0x11C:0x128; RO)

The Header Log register captures the header for the transaction that generated an error. This register is 16 bytes in length.

Bit Location	Attribute	Default Value	Description
127:0	ROS	0b	Header of the packet in error (TLP or DLLP).

9.5.2 Serial Number

Note: 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. This capability is enabled by the "Serial Number enable" flag in the NVM.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x140	Next Capability Ptr. 0x1C0	Version (0x1)	Serial ID Capability ID (0x0003)	
0x144	Serial Number Register (Lower Dword)			
0x148	Serial Number Register (Upper Dword)			

9.5.2.1 Device Serial Number Enhanced Capability Header (0x140; RO)

The following table lists the allocation of register fields in the device serial number enhanced capability header. It also lists the respective bit definitions. The extended capability ID for the device serial number capability is 0x0003.

Bit(s) Location	Default value	Attributes	Description
15:0	0x0003	RO	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	0x1	RO	Capability Version This field is a PCI-SIG defined version number that indicates the version of the current capability structure.
31:20	0x1C0	RO	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.

9.5.2.2 Serial Number Register (0x144:0x148; RO)

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit extended unique identifier (EUI-64™). Table 9-13 lists the allocation of register fields in the Serial Number register. Table 9-13 also lists the respective bit definitions.

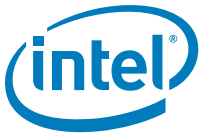
Table 9-13. Serial Number Register

31:0
Serial Number Register (Lower Dword)
Serial Number Register (Upper word)
63:32

Serial number definition in FoxvillePENTIUM® PRO FAMILY DEVELOPER'S MANUAL, VOLUME 2:

Table 9-14. SN Definition

Bit(s) Location	Attributes	Description
63:0	RO	PCIe Device Serial Number This field contains the IEEE defined 64-bit extended unique identifier (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.



Serial number uses the MAC address according to the following definition:

Field	Extension identifier					Company ID		
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 MAC address in the following form:

Field	Extension identifier			MAC Label		Company ID		
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

The MAC label in this case is 0xFFFF.

For example, assume that the company ID is (Intel) 00-A0-C9 and the extension identifier is 23-45-67. In this case, the 64-bit serial number is:

Field	Extension identifier			MAC Label		Company ID		
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7
	67	45	23	FF	FF	C9	A0	00
	Most significant byte							Least significant byte
	Most significant bit							Least significant bit

The MAC address is the MAC address as loaded from the NVM into the RAL and RAH registers.

The translation from NVM words 0 to 2 to the serial number is as follows:

- Serial number ADDR+0 = NVM byte 5
- Serial number ADDR+1 = NVM byte 4
- Serial number ADDR+2 = NVM byte 3
- Serial number ADDR+3 and 4 = 0xFF 0xFF
- Serial number ADDR+5 = NVM byte 2
- Serial number ADDR+6 = NVM byte 1
- Serial number ADDR +7 = NVM byte 0

The official document defining EUI-64 is: http://standards.ieee.org/regauth/oui/tutorials/EUI64.html

9.5.3 Latency Tolerance Requirement Reporting (LTR) Capability

The PCI Express Latency Tolerance Requirement Reporting Capability is an optional Extended Capability that allows software to provide platform latency information to devices with upstream ports (Endpoints and Switches). This capability structure is required if the device supports Latency Tolerance Requirement Reporting (LTR). This capability is enabled by the "LTR_EN" flag in the NVM.

The following table lists the PCIe LTR extended capability structure for PCIe devices.

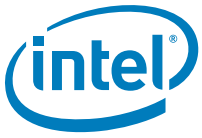
Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x1C0	Next Capability Ptr. (0x1F0)	Version (0x1)	LTR Capability ID (0x18)	
0x1C4	Maximum Non-Snooped Platform Latency Tolerance Register		Maximum Snooped Platform Latency Tolerance Register	

9.5.3.1 LTR CAP ID (0x1C0; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x18	LTR Capability ID PCIe extended capability ID indicating LTR capability.
19:16	RO	0x1	Version Number PCIe LTR extended capability version number.
31:20	RO	0x1F0	Next Capability Pointer

9.5.3.2 LTR Capabilities (0x1C4; RW)

Bit Location	Attribute	Default Value	Description
9:0	RW	0x0	Maximum Snoop Latency Value Along with the Max Snoop Latency Scale field, this register specifies the maximum nosnoop latency that a device is permitted to request. Software should set this to the platform's maximum supported latency or less. Field is also an indicator of the platforms maximum latency, should an endpoint send up LTR Latency Values with the Requirement bit not set.
12:10	RW	0x0	Max Snoop Latency Scale This field provides a scale for the value contained within the Maximum Snoop Latency Value field. Encoding: 000 – Value times 1ns 001 – Value times 32ns 010 – Value times 1,024ns 011 – Value times 32,768ns 100 – Value times 1,048,576ns 101 – Value times 33,554,432ns 110-111 – Not Permitted
15:13	RO	0x0	Reserved



Bit Location	Attribute	Default Value	Description
25:16	RW	0x0	Max No-Snoop Latency Value Along with the Max No-Snoop Latency Scale field, this register specifies the maximum no-snoop latency that a device is permitted to request. Software should set this to the platform's maximum supported latency or less. Field is also an indicator of the platforms maximum latency, should an endpoint send up LTR Latency Values with the Requirement bit not set.
28:26	RW	0x0	Max No-Snoop Latency Scale — This register provides a scale for the value contained within the Maximum Non-Snoop Latency Value field. Encoding: 000 – Value times 1 ns 001 – Value times 32 ns 010 – Value times 1,024 ns 011 – Value times 32,768 ns 100 – Value times 1,048,576 ns 101 – Value times 33,554,432 ns 110-111 – Not Permitted
31:29	RO	0x0	Reserved.

9.5.4 L1 Sub States Capability (offset - 0x1E0)

The L1 substates capabilities define and optional mechanism of substates of the L1 Link state, which allow lower idle power of the system. The new L1 PM Substates are applicable in both the ASPM and PCI-PM L1 Link states. L1 PM Substate management utilizes a per-Link sideband signal - CLKREQ#. The L1 substate capability in the PCIe configuration space is enabled by the "L1 Substate Enable" flag in the "PCIe L1 Substates Capability" word in the NVM.

Byte offset / Bits	bits 31:20	bits 19:16	bits 15:0
0x1E0	Next Capability Ptr - (0x0000)	Version (0x1)	L1 Sub States Capability ID (0x1E)
0x1E4	L1 PM Sub states Capabilities Register		
0x1E8	L1 PM Sub states Control 1 Register		
0x1EC	L1 PM Sub states Control 2 Register		

9.5.4.1 L1 Sub States Capability Header (0x1E0; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x1E	L1 Sub States Capability ID
19:16	RO	0x1	Version Number
31:20	RO	0x0000	Next Capability Pointer (0x000 for last capability structure)

9.5.4.2 L1 PM Sub States Capabilities Register (0x1E4; RW)

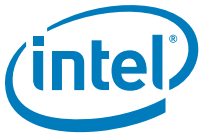
Bit Location	Attribute	Default Value	Description
0	RO	1	PCI-PM L1.2 Supported
1	RO	1	PCI-PM L1.1 Supported
2	RO	1	ASPM L1.2 Supported
3	RO	1	ASPM L1.1 Supported
4	RO	1	L1 PM Substates Supported
5:7	RSVDP	0	Reserved
8:15	RO	0x37	T-Common-mode - Time (in µs) required for this Port to re-establish common mode (0-255 µs), when exiting L1.2.
16:17	RO	0	Port T_POWER_ON Scale - Specifies the scale used for the Port T_POWER_ON Value field in the L1 PM Substates Capabilities register: 00b = 2µs 01b = 10usec 10b = 100 usec 11b = reserved
18	RSVDP	0	Reserved
19:23	HwInit	0x9	Port T_POWER_ON Value - the time (in us) that this Port requires from the port on the opposite side of Link to wait in PLL Warmup + L1.2.Exit after sampling CLKREQ# before actively driving the interface. The value of Port T_POWER_ON is calculated by multiplying the value in this field by the scale value in the Port T_POWER_ON Scale Note: The "PLL Warmup" parameter is defined in the PCIE_L1_EXTCLK register. The programmed T_POWER_ON must be large enough so that the L1.2.Exit is guaranteed to be equal or greater than 16usec.
24:31	RSVDP	0	Reserved

9.5.4.3 L1 PM Substates Control 1 Register (0x1E8; RW)

Bit Location	Attribute	Default Value	Description
0	RW	0	PCI-PM L1.2 Enable
1	RW	0	PCI-PM L1.1 Enable
2	RW	0	ASPM L1.2 Enable
3	RW	0	ASPM L1.1 Enable
4:7	RsvdP	0	Reserved
8:15	RW	0	Common Mode Restore Time for downstream port (n/a for NIC) – Sets value of T-common-mode (in µs). This field may be modified only when the ASPM L1.2 Enable and PCI-PM L1.2 Enable bits are both Cleared.
16:25	RW	0	LTR_L1.2_THRESHOLD_Value -indicates the LTR threshold used to determine if entry into L1 results in L1.1 (if enabled) or L1.2 (if enabled). This field may be modified only when the ASPM L1.2 Enable bit is Cleared.
26:28	RsvdP	0	Reserved
29:31	RW	0	LTR_L1.2_THRESHOLD_Scale - provides a scale for the value contained within the LTR_L1.2_THRESHOLD_Value. Permitted values are between 000b and 101b. This field may be modified only when the ASPM L1.2 Enable bit is Cleared.

9.5.4.4 L1 PM Sub states Control 2 Register (0x1EC; RW)

this register may be modified only when both ASPM L1.2 Enable bit and PCI-PM L1.2 Enable bit are both cleared.



Bit Location	Attribute	Default Value	Description
0:1	RW	0	T_POWER_ON Scale - Specifies the scale used for T_POWER_ON Value: 00b = 2µs 01b = 10µs 10b = 100µs 11b = Reserved
2	RsvdP	0	reserved
3:7	RW	~~0x8~~ 0x9	T_POWER_ON Value - the minimum amount of time (in µs) that the Port must wait in L1.2. Exit after sampling CLKREQ# before actively driving the interface. T_POWER_ON is calculated by multiplying the value of this field by T_POWER_ON Scale. Loaded from the "Port T_POWER_ON" field in the NVM
8:31	RsvdP	0	Reserved

9.5.5 Precision Time Measurements Capability - PTM (offset - 0x1F0)

The PCIe PTM capability is an optional extended capability supporting precision time measurements. The PTM capability in the PCIe configuration space is enabled by the "PTM Enable" flag in the "PCIe PTM Control" word in the NVM.

Byte offset / Bits	bits 31:20	bits 19:16	bits 15:0
0x1F0	Next Capability Ptr - (0x1E0)	Version (0x1)	PTM Capability ID (0x1F)
0x1F4	PTM Capabilities Register		
0x1F8	PTM Control Register		

9.5.5.1 PTM Capability Header (0x1F0; RO)

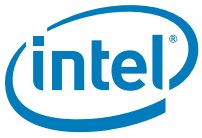
Bit Location	Attribute	Default Value	Description
15:0	RO	0x1F	Precision Time Measurements Capability ID
19:16	RO	0x1	Version Number
31:20	RO	0x1E0	Next Capability Pointer

9.5.5.2 PTM Capability Register (0x1F4; RO)

Bit Location	Attribute	Default Value	Description
0	RO	1	PTM Requester Capable - Endpoints and Root Complex implement the PTM Requester role. Loaded from the "PTM Requester Capable" flag in the NVM (word 0x0049).
1	RO	0	PTM Responder Capable
2	RO	0	PTM Root Capable
3:7	RSVD	0x0	Reserved
8:15	RO	0x4	Local Clock Granularity - Indicates the period of this Time Source's local clock in nsec. Loaded from the "PTM Effective Granularity" field in the NVM (word 0x0049).
16:31	RsvdP	0	Reserved

9.5.5.3 PTM Control Register (0x1F8; RW)

Bit Location	Attribute	Default Value	Description
0	RW	0	PTM capability Enabled
1	RO	0	Root Select
2:7	RsvdP	0	Reserved
8:15	RW	0	Effective Granularity - Provides and information of the expected PTM accuracy in nsec.
16:31	RsvdP	0	Reserved



10.0 System Manageability

This chapter describes the "Manageability Host Interface" including a whole list of the "Host Interface Commands". A mechanism to isolate the host in the case of malfunction software is described in "Host Isolate Support".

10.1 Manageability Host Interface

This section details host interaction with the manageability portion of Foxville. The software device driver communicates with the manageability block through CSR access. The manageability is mapped to address space 0x8800 to 0x8FFF on the slave bus of the function:

- Control registers are located at address 0x8F00.
- Direct access to the internal ARC data RAM: The internal shared (between Firmware and Software) RAM is mapped to address space 0x8800 to 0x8EFF. Writing/reading to this address space goes directly to the RAM.

10.1.1 Host Slave Command Interface - Event Flow

This interface is used for the external host software to access the manageability subsystem. Host software writes a command block or read data structure directly from the data RAM. Host software controls these transactions through a slave access to the control register.

The following flow shows the process of initiating a command to the manageability block:

1. Software clears the *FWSTS.FWRI* flag (clear by write one) avoiding any previous residuals.
2. Software takes ownership of the Management Host interface using the flow described in Section 4.8.1.
3. Software reads the *HOST Interface Control* Register checking for active *Enable* (*HICR.En*) bit.
4. Software writes the relevant command block into the shared ARC RAM area (addresses 0x8800...)
5. Software sets the *Command* (*HICR.C*) bit in the *HOST Interface Control* Register.
6. Software checks the *FWSTS.FWRI* flag to make sure a firmware reset didn't occur during the command processing. If this bit is set, the command may have failed.
7. Software polls the command bit (*HICR.C*) to be cleared by Foxville.
8. When Foxville completes the command processing, it clears the *Command* bit (*HICR.C*). If the command includes some data, Foxville clears the command bit only after the data is placed in the shared RAM. If the Software reads the *HOST Interface Control* register and the *HICR.SV* bit is set to 1b, then there is a valid status of the last command in the shared RAM. If the *HICR.SV* bit is not set, then the command has failed with no status in the RAM.

On the completion of access to the shared RAM, the Software should release ownership of the shared RAM using the flow described in Section 4.8.2.

10.1.2 Host Interface Structure

Command Structure: The Table 10-1 below describes the command structure programmed by the Software in the shared ARC RAM (address 0x8800...).

Table 10-1. Host Driver Command Structure

#Byte	Description	Bit	Value	Description
0	Command	7:0	Command Dependent	Specifies which host command to process.
1	Buffer Length	7:0	Command Length	Command Data Buffer length: 0 to 252, not including 32 bits of header.
3	Checksum	7:0	Defined Below	Checksum signature.
255:4	Data Buffer	7:0	Command Dependent	Command Specific Data Minimum buffer size: 0. Maximum buffer size: 252.

Response / Status Structure: The Table 10-2 below describes the structure used by Foxville providing a status response to the Software. The status is posted also in the shared ARC RAM.

Table 10-2. Status Structure Returned to Host Driver

#Byte	Description	Bit	Value	Description
0	Command	7:0	Command Dependent	Command ID.
1	Buffer Length	7:0	Status Dependent	Status buffer length: 252:0
2	Return Status	7:0	Depends on Command Executing Results	0x1 Status OK 0x2 Illegal command ID 0x3 Unsupported command 0x4 Illegal payload length 0x5 Checksum failed 0x6 Data Error 0x7 Invalid parameter 0x8 - 0xFF Reserved
3	Checksum	7:0	Defined Below	Checksum signature.
255:4	Data Buffer		Status Dependent	Status configuration parameters Minimum Buffer Size: 0. Maximal Buffer Size: 252.

Checksum Calculation: The Host Command/Status structure is summed with this field cleared to 0b and then posting the negative result to this field.

10.1.3 Host Interface Commands

The Table 10-3 below summarizes the existing host slave commands:

Table 10-3. Host Slave Commands

Command Opcode	Description
0x3	Get Firmware Version
0xF0	Write Configuration
0xF1	Read Configuration
~~0x30~~0x30	Set PM capabilities
0x31	Set Filter Indirect Table Select

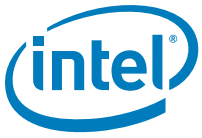


Table 10-3. Host Slave Commands

0xEA	Get Firmware Proxying Capabilities
0xEB	Set Firmware Proxying Configuration
0x77	Set ARP Proxy Table Entry
0x78	Set NS (Neighbor Solicitation) Proxy Table Entry
0x79	Set mDNS Proxy
0x80	Imaging Wakeup Configuration Command

10.1.3.1 Get Firmware Version Host Command

This command is used for aliveness checking, and to receive more information about HW version ROM version and Patch version.

Table 10-4. Commands to Get Firmware version

Byte	Name	Bit	Value	Description
0	Command	7:0	0x3	Get Firmware version command.
1	Buffer Length	7:0	0x0	No data bytes attached to this command.
2		7:0	0x0	
3	Checksum	7:0		Checksum signature of the Host command.

Following is the status returned on this command:

Table 10-5. Get Firmware version Status Returned

Byte	Name	Bit	Value	Description
0	Command	7:0	0x03	Get Firmware version command.
1	Buffer Length	7:0	0xB	11 bytes attached to this status.
2	Status	7:0		See Table 10-2
3	Checksum	7:0		Checksum signature of the Host command.
4	ROM SVN Version (MSB)	7:0		
5	ROM SVN Version	7:0		
6	ROM SVN Version (LSB)	7:0		
7	ROM Release - major	7:0		
8	ROM Release - minor	7:0		
9	NVM/RAM SVN version (MSB)	7:0		
10	NVM/RAM SVN version	7:0		
11	NVM/RAM SVN version (LSB)	7:0		
12	NVM/RAM release - major	7:0		
13	NVM/RAM release - minor	7:0		
14	Silicon version	7:0		If bit 7 is set, indicates an FPGA version.

10.1.3.2 Write Configuration Host Command

This command is used to allow debug access to CSRs or FW registers. This command is available only in debug versions of the firmware.

Table 10-6. Write Configuration Command

Byte	Name	Bit	Value	Description
0	Command	7:0	0xF0	Write Configuration command.
1	Buffer Length	7:0	0xA	10 data bytes attached to this command.
2		7:0	0x0	
3	Checksum	7:0		Checksum signature of the Host command.
4	Address Space Selector	7:0		0x0: CSR space 0x1: Aux Space 0x2: Internal Management Memory Space (ARC memory) 0x3:0xFF: Reserved.
5	Reserved	7:0		Reserved
6:9	Address	7:0		The address to write in the selected space
10:13	Data	7:0		The data to write

Following is the status returned on this command:

Table 10-7. Write Configuration Status Returned

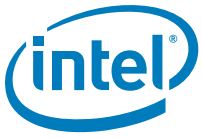
Byte	Name	Bit	Value	Description
0	Command	7:0	0xF0	Write Configuration command.
1	Buffer Length	7:0	0x0	No bytes attached to this status.
2	Status	7:0		See Table 10-2
3		7:0		Checksum signature of the Host command.

10.1.3.3 Read Configuration Host Command

This command is used to allow debug access to CSRs or FW registers. This command is available only in debug versions of the firmware.

Table 10-8. Read Configuration Command

Byte	Name	Bit	Value	Description
0	Command	7:0	0xF1	Read Configuration command.
1	Buffer Length	7:0	0x6	6 data bytes attached to this command.
2		7:0	0x0	
3	Checksum	7:0		Checksum signature of the Host command.
4	Address Space Selector	7:0		0x0: CSR space 0x1: Aux Space 0x2: Internal Management Memory Space (ARC memory) 0x3:0xFF: Reserved.
5	Reserved	7:0		Reserved
6:9	Address	7:0		The address to read from the selected space



Following is the status returned on this command:

Table 10-9. Read Configuration Status Returned

Byte	Name	Bit	Value	Description
0	Command	7:0	0xF1	Read Configuration command.
1	Buffer Length	7:0	0x4	4 bytes attached to this status.
2	Status	7:0		See Table 10-2
3		7:0		Checksum signature of the Host command.
4:7	Read Data	7:0		The data read from the CSR/memory requested

10.1.3.4 Set PM Capabilities

This command is used for setting power management capabilities.

Table 10-10. SetPM capabilities Host Slave Interface Command Structure

Byte	Name	Bit	Value	Description
0	Command	7:0	0x30	Driver info command.
1	Buffer Length	7:0	0x1	Port Number + 4 bytes of the Driver info
2	Reserved	7:0	0x0	Reserved
3	Checksum	7:0		Checksum signature of the Host command.
4	PM Settings	7:0	See description	bit 0: Enable Critical Session Mode (Keep_PHY_Link_Up and Veto Bit) 0x0 - Disabled 0x1 - Enabled When critical session mode is enabled, the following behaviors are disabled: • The PHY is not reset on PE_RST# 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. bit 1: Wake on MagicPacket inD3cold is required. 0x0 - Disabled 0x1 - Enabled Host driver may be requesting it before entering D3. bit 2: ULP enable in D3cold. 0x0 - Disabled 0x1 - Enabled bits 3-7: Reserved

Table 10-11. SetHost Slave Interface Status

Byte	Name	Bit	Value	Description
0	Command	7:0	0x30	Driver Info command
1	Buffer Length	7:0	0x0	No data in return status
2	Return Status	7:0	Status	Status
3	Checksum	7:0		Checksum signature

10.1.3.5 Set Filter Indirect Table Select

This command is used to program the FHFTSL register that is required to select a group of 8 flexible filters out of the existing 32 filters. The flexible filters are accessible by the FHFT and FHFT_EXT registers in the CSR space. These filters are enabled by the WUFC and WUFC_EXT registers in the CSR space as well.

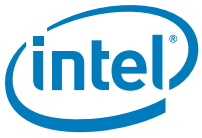


Table 10-12. Set Filter Indirect Table Select Command Structure

Byte	Name	Bit	Value	Description
0	Command	7:0	0x31	Command Opcode
1	Buffer Length	7:0	0x1	Command Buffer Length
2	Reserved	7:0	0x0	Reserved
3	Checksum	7:0		Checksum signature of the Host command.
4	Filter Group Select	7:0	See description	0x0: Select filters 0-7 0x1: Select filters 15-8 0x2: Select filters 23-16 0x3: Select filters 31-24 Else: Reserved

Table 10-13. Set Filter Indirect Table Select Status

Byte	Name	Bit	Value	Description
0	Command	7:0	0x31	Driver Info command
1	Buffer Length	7:0	0x0	No data in return status
2	Return Status	7:0	Status	Status
3	Checksum	7:0		Checksum signature

10.1.3.6 Host Proxying Commands

Software Device driver will send to Firmware via shared RAM interface the following Proxying commands, using the interface described in Section 10.1.1:

1. Get Firmware Proxying Capabilities Command (See Table 10-14) to receive information on Protocol offloads supported.
2. Set Firmware Proxying Configuration Command (See Table 10-16) to define the required proxying behavior.
3. Send the required Proxying information for the Protocol offloads supported by Firmware via the following commands:
   a. Set ARP Proxy Table Entry (See Table 10-18).
   b. Set NS (Neighbor Solicitation) Proxy Table Entry (See Table 10-20).

Following the reception of the commands, Firmware will acknowledge execution of the command via the shared RAM interface using the following responses according to the command issued:

1. Get Firmware Proxying Capabilities Response (See Table 10-15).
2. Set Firmware Proxying Configuration Response (See Table 10-17).
3. Acknowledge reception of Proxying information via the following responses:
   a. Set ARP Proxy Table Entry Response (See Table 10-19).
   b. Set NS (Neighbor Solicitation) Proxy Table Entry Response (See Table 10-21)

10.1.3.6.1 Get Firmware Proxying Capabilities

This command is used to provide the driver information on protocol offload types supported by Foxville.

Table 10-14. Get Firmware Proxying Capabilities Command

Byte	Name	Bit	Value	Description
0	Command	7:0	0xEA	GET Firmware Proxying Capabilities
1	Buffer length	7:0	0x2	
2	Reserved	7:0	0x0	Must be zeroed by host
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the command is targeted at.
5	Page	7:0	0x1	Get capabilities page number If response exceeds 256 bytes including header (Maximum page size), Software should issue multiple Get Firmware Proxying Capabilities commands with increasing page number until a response with buffer length smaller than 252 is received or a response with a Status field with an Unsupported Page Number is received. ***Note:*** Maximum Page size is 256 Bytes including Header information.

Firmware returns the following status for this command:

Note: The Firmware status reply includes a series of two values

{Protocol offload capability type and version, number of entries for this type of Protocol offload}

Currently the following capabilities are defined, ARP proxy, NS proxy, MLD proxy and mDNS proxy. If the structure is too big to transfer in one time the driver can ask for additional pages by incrementing the page field.

Table 10-15. Get Firmware Proxying Capabilities Response

Byte	Name	Bit	Value	Description
0	Command	7:0	0xEA	Get Firmware Proxying Capabilities
1	Buffer length	7:0	0xB	The buffer length can vary according to the capabilities the FW supports i.e. if the FW supports 4 capabilities it would be 0xB for less capabilities 2 bytes are reduced per capability
2	Return Status	7:0	0x1	0x0 - Unsupported Page number 0x1 - Status OK 0x2 to 0xFF - Error
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the response is for.
5	Page	7:0	0x1	First page of capabilities
6	Total Cap size	7:0	0x8	Size of capability structure in bytes
7	ARP proxy version 1	7:0	0x1	
8	Number of ARP entries	7:0	Number of entries	Number of ARP entries supported
9	NS proxy version 1	7:0	0x2	
10	Number of NS proxy entries	7:0	Number of entries	Number of NS entries supported
11	MLD proxy version	7:0	0x3	

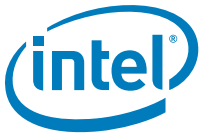


Table 10-15. Get Firmware Proxying Capabilities Response

12	MLD support	7:0	Version of MLD supported	0x0 - not supported 0x1 - MLD version 1 compatibility mode 0x2 - MLD version 2 compatibility mode 0x3 - Both versions supported 0x4 - x0FF: Reserved
13	mDNS proxy version	7:0	0x4	
14	mDNS proxy offload support	7:0	Data buffer Size	Size of data buffer allocated for mDNS proxy offload data in KB

10.1.3.6.2 Set Firmware Proxying Configuration

This command is used to provide information to Firmware on how to implement protocol offloads supported by Foxville. The Firmware Proxying Configuration command includes a series of two values: {Command type and version, Command Data for this type of command}. Currently only one Configuration command is defined:

1. No Match - Command defines expected behavior when receiving a Proxying packet that's not supported.

Foxville keeps its proxy settings after D3 to D0 transition, until they are disabled by host. However, the proxy filters are reset by PE_RST_N assertion/deassertion that occurs when transiting from D3 to D0 via Dr state. Therefore, FW is required to reconfigure the proxy filters after PE_RST_N deassertion, in the aim to maintain proxying until the driver is up again. To avoid a race condition between FW and SW while accessing proxy registers, the driver should first disable proxying via the host interface before modifying the proxy registers.

If the structure is too big to transfer in one time the driver can ask for additional pages by incrementing the page field.

Table 10-16. Set Firmware Proxying Configuration Command

Byte	Name	Bit	Value	Description
0	Command	7:0	0xEB	Set Firmware Proxying Configuration
1	Buffer length	7:0	0x6	
2	Reserved	7:0	0x0	Must be zeroed by host
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the command is targeted at. ***Note:***
5	No Match	7:0	0x1	No Match command Defines how Firmware handles unsupported proxying packets.
6	No Match data	7:0	0x0 or 0x1	No Match data 0x0 - Discard unsupported proxying packets 0x1 - Issue Wake on reception of unsupported packets. ***Note:*** 0x0 is the default value if no configuration command is issued.
7	Reserved	7:0	0x0	Reserved
8	Reserved	7:0	0x0	Reserved
9	Enable MLD	7:0	0x0, 0x1 or 0x2	0x0: Do not enable MLD 0x1: Enable MLD version 1 0x2: Enable MLD version 2 0x3 - x0FF: Reserved

The Firmware returns the following Response for this command:

Table 10-17. Set Firmware Proxying Configuration Response

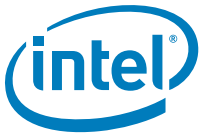
Byte	Name	Bit	Value	Description
0	Command	7:0	0xEB	Get Firmware Proxying Capabilities
1	Buffer length	7:0	0x6	
2	Return Status	7:0	0x1	0x0 - Undefined Error 0x1 - Status OK 0x2 - Unsupported command 0x3 - Checksum Error 0x4 - Buffer Length Error 0x5 to 0xFF - Error
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the status is from.
5	No Match	7:0	0x1	No Match command Defines how Firmware handles unsupported proxying packets.
6	No Match data	7:0	0x0 or 0x1	No Match Data 0x0 - Discard unsupported proxying packets 0x1 - Issue Wake on reception of unsupported packets. ***Note:*** 0x0 is the default value if no configuration command is issued.
7	Reserved	7:0	0x0	Reserved
8	Reserved	7:0	0x0	Reserved
9	Enable MLD	7:0	0x0, 0x1 or 0x2	Copied from the Set Firmware Proxying Configuration Command

10.1.3.6.3 Set ARP Proxy Table Entry

This command structure is described in Table 10-18. Activating an entry the software driver should post the command with *Active* field set to 0x1. To disable an entry the software should post this entry with the *Active* field cleared to 0x0. The software must initially disable all unused entries.

Table 10-18. Set ARP Proxy Table Entry Command

Byte	Name	Bit	Value	Description
0	Command	7:0	0x77	Set ARP proxy command
1	Buffer length	7:0	0x13	
2	Reserved	7:0	0x0	Must be zeroed by host
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the command is targeted at. ***Note:***
5	Sub command	7:0	0x3	Set proxy capabilities
6	ARP proxy version 1	7:0	0x1	ARP version 1 entry



Byte	Name	Bit	Value	Description
7	Table index	7:0	Index	Table index Each Set proxy command is held in a separate Table index. Field defines Table Index for current command. ***Notes:*** 1. Table Index values begin at 1. 2. Only a single ARP proxy table entry is supported and only a Table index value of 1 is valid. 3. To change contents of a table entry the relevant Table index should be invalidated (Write command to the Table index with Active field = 0x0) before writing new content.
8	Active	7:0	0x1 or 0x0	Set to 0x1 to activate table index Set to 0x0 to invalidate it If set to 0, values of all following fields are ignored
14:9	MAC Address	7:0		MAC Address to reply to ARP request
18:15	Local IP Address	7:0		Local IP Address of station
22:19	Remote IP Address	7:0		Remote IP Address A value of 0x0 indicates any remote IP address

The Firmware returns the following status for this command:

Table 10-19. Set ARP Proxy Table Entry Response

Byte	Name	Bit	Value	Description
0	Command	7:0	0x77	Set ARP proxy command
1	Buffer length	7:0	0x13	
2	Status	7:0	0x1	0x0 - Unsupported Table Index 0x1 - Status OK 0x2 - Table Index in use. 0x3 - Unsupported command 0x4 - Checksum Error 0x5 - Buffer Length Error 0x6 to 0xFF - Error
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the response is for.
5	Sub command	7:0	0x3	Set proxy capabilities
6	ARP proxy version 1	7:0	0x1	ARP version 1 entry
7	Table index	7:0	Index	
8	Active	7:0	0x1 or 0x0	Set if entry is active
14:9	MAC Address	7:0		
18:15	Local IP Address	7:0		
22:19	Remote IP Address	7:0		

10.1.3.6.4 Set NS (Neighbor Solicitation) Proxy Table Entry

This command structure is described in Table 10-20. To set an entry the Software should post the command with *Active* field set to 0x1. To disable an entry the software should post this entry with the *Active* field cleared to 0x0. The software must initially disable all unused entries.

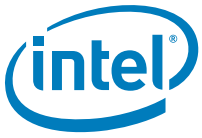
Table 10-20. Set NS Proxy Table Entry Command

Byte	Name	Bit	Value	Description
0	Command	7:0	0x78	Set NS proxy command
1	Buffer length	7:0	0x4B	
2	Reserved	7:0	0x0	Must be zeroed by host
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the command is targeted at. ***Note:***
5	Sub command	7:0	0x3	Set proxy capabilities
6	NS proxy version 1	7:0	0x2	NS version 1 entry
7	Table index	7:0	Index	Table index Each Set proxy command is held in a separate Table index. Field defines Table Index for current command. ***Notes:*** 1. Table Index values begin at 1. 2. Up to two NS proxy table entries are supported and only Table index values of 1 and 2 are valid. 3. To change contents of a table entry the relevant Table index should be invalidated (Write command to the Table index with Active field = 0x0) before writing new content.
8	Active	7:0	0x1 or 0x0	Set to 0x1 to activate table index Set to 0x0 to invalidate it If set to 0, values of all following fields are ignored
14:9	MAC Address	7:0		MAC Address
30:15	Local IPv6 Address 1	7:0		Local IPv6 Address 1. This address is used by firmware for NS proxying.
46:31	Local IPv6 Address 2	7:0		Local IPv6 Address 2 If there is only one local address value placed here is 0x0. When not 0x0, this address is used by firmware for NS proxying.
62:47	Remote IPv6 Address	7:0		Remote IPv6 Address A value of 0x0 indicates any address.
78:63	Solicited IPv6 Address	7:0		Solicited IPv6 Address. This address is used by firmware for MLD proxying.

The Firmware returns the following status for this command:

Table 10-21. Set NS Proxy Table Entry Response

Byte	Name	Bit	Value	Description
0	Command	7:0	0x78	Set NS proxy command
1	Buffer length	7:0	0x4B	
2	Status	7:0	0x1	0x0 - Unsupported Table Index 0x1 - Status OK 0x2 - Table Index in use. 0x3 - Unsupported command 0x4 - Checksum Error 0x5 - Buffer Length Error 0x6 to 0xFF - Error
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the status is for.



Byte	Name	Bit	Value	Description
5	Sub command	7:0	0x3	Set proxy capabilities
6	NS proxy version 1	7:0	0x2	NS version 1 entry
7	Table index	7:0	Index	
8	Active	7:0	0x1 or 0x0	Set if entry is active
14:9	MAC Address	7:0		MAC Address
30:15	Local IPv6 Address 1	7:0		Local IPv6 Address 1
46:31	Local IPv6 Address 2	7:0		Local IPv6 Address 2 If there is only one local address value placed is 0x0
62:47	Remote IPv6 Address	7:0		Remote IPv6 Address A value of 0x0 indicates any address.
78:63	Solicited IPv6 Address	7:0		Solicited IPv6 Address

10.1.3.6.5 Set mDNS Proxy

To enable mDNS proxy the software needs to handoff the mDNS data to the FW. The software generally pass the mDNS data in chunks of 128 bytes using the Set mDNS Proxy command. However, in Foxville, for saving on-die memory space, the mDNS data is loaded by the software directly into the flash area provisioned for it. The exact structure of the mDNS data is specified in the mDNS Proxy SAS document. The command is issued by the software whenever the system goes to sleep, after the mDNS data is updated in the flash if necessary.

Table 10-22. Set mDNS Proxy Command

Byte	Name	Bit	Value	Description
0	Command	7:0	0x79	Set mDNS Proxy Command
1	Buffer length	7:0	0x6	
2	Reserved	7:0	0x0	Must be zeroed by host
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	0x0	Indicates the port number that the command is targeted at.
5	Sub command	7:0	0x3	Set proxy capabilities
6	mDNS proxy version 1	7:0	0x4	mDNS version 1 entry
7	DataChunk index	7:0	0x0	Data Chunk index is used to pass all mDNS proxy data in chunks starting from Chunk[0] till the last chunk
8	Active	7:0	0x0	Bit 0: Set to 1b to activate chunk index Set to 0b to invalidate it If set to 0, values of all following fields are ignored Bit 1: Set to 1b to indicate last chunk Set to 0b to indicate other (non last) chunk
9	Chunk Data Length	7:0	0x0	Set to 0x80 for non last data chunks, for last data chunk indicates the number of valid bytes in the chunk
137:10	mDNS Data Chunk	7:0		mDNS Data Chunk 128 bytes per chunk, last chunk valid bytes are defined by Data Length and padded with zeros up to 128 bytes. ***Note:*** This field is not present in

The Firmware returns the following status for this command:

Table 10-23. Set mDNS Proxy Response

Byte	Name	Bit	Value	Description
0	Command	7:0	0x79	Set mDNS Proxy Response
1	Buffer length	7:0	0x6	
2	Status	7:0	0x1	0x0 - Unsupported Table Index 0x1 - Status OK 0x2 - Table Index in use. 0x3 - Unsupported command 0x4 - Checksum Error 0x5 - Buffer Length Error 0x6 to 0xFF - Error
3	Checksum	7:0		Checksum signature
4	Port Number	7:0	Port Number	Indicates the port number that the status is for.
5	Sub command	7:0	0x3	Set proxy capabilities
6	mDNS proxy version 1	7:0	0x4	mDNS version 1 entry
7	DataChunk index	7:0	Index	Data Chunk index is used to pass all mDNS proxy data in chunks starting from Chunk[0] till the last chunk
8	Active	7:0		Reflection of the Active byte from the mDNS proxy command
9	Chunk Data Length	7:0		Reflection of the Chunk Data Length byte from the mDNS proxy command

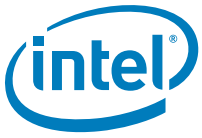
10.1.3.7 Imaging Wakeup Configuration Host Command

This command is used to activate some of the wakeup filtering patterns relative to the printing/imaging market. It uses internal MDEF filtering capabilities to be completed by extra filtering performed in FW.

Table 10-24. Imaging Wakeup Configuration Host Command

Byte	Name	Bit	Value	Description
0	Command	7:0	0x80	Imaging wakeup configuration command.
1	Buffer Length	7:0	0x22	2 or 34 bytes needed for the configuration, according to the wakeup type(s) activated (see in byte 5).
2	Reserved	7:0	0x0	Reserved.
3	Checksum	7:0		Checksum signature of the Host command.
4	Port Number	7:0	0x0	Always 0x0 in Foxville.
5	Wakeup Types	7:0		Bitmap to set the wakeup type(s) to be activated: Bit 0 = NetBIOS name query wakeup enabled/disabled. If set to 1b, a 32-bytes data buffer is included in the command. Bit 1 = SNMP broadcast query wakeup enabled/disabled. Bit 2 = SLP query wakeup enabled/disabled. Bit 3 = SSDP query wakeup enabled/disabled. Bit 7:4 = Reserved (must be 0x0).
38:6	Data Buffer (optional)	7:0	NetBIOS Name String	The 32 bytes string included in the name query packet. For example, after it has been encoded as required by the NetBIOS protocol.

Following is the status returned by this command:



Byte	Name	Bit	Value	Description
0	Command	7:0	0x80	SNMP broadcast query wakeup configuration command.
1	Buffer Length	7:0	0x0	No data in return status.
2	Return Status	7:0		0x1 = Status OK. 0x3 = Unsupported command, returned if the PI features are disabled in NVM.
3	Checksum	7:0		Checksum signature.

10.1.4 Host Isolate Support

The CSME might conclude that it is prevented from accessing the network due to malfunction software. In such a case, it is capable to isolate Foxville from the host interface, disabling any access to the device by any software. This is done using the TCO reset command.

If TCO isolate is enabled in the NVM, The TCO Isolate command disables any PCIe write operations to the device. As the driver needs to access the CSR space in order to provide descriptors to the NIC, this operation will also stop the network traffic including OS2BMC and CSME-to-OS traffic as soon as the existing transmit and receive descriptor queues are exhausted.