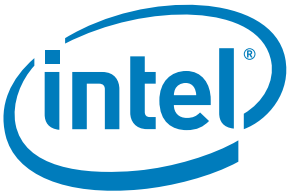


Intel® Ethernet Controller I210-CS/ CL Datasheet

Ethernet Networking Division (ND)

Features:

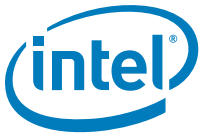
- Small package: 9 x 9 mm
- PCIe v2.1 (2.5 GT/s) x1, with Switching Voltage Regulator (iSVR)
- Integrated Non-Volatile Memory (iNVM)
- I210-CS: AEC-Q100 grade 3 certified (-40 °C to 85 °C) with less than or equal to 500 DPM
- I210-CL: AEC-Q100 grade 3 certified (-40 °C to 85 °C) with less than or equal to 20 DPM
- I210-CL: AEC-Q100 Rev H certified
- Platform Power Efficiency
  - Proxy: ECMA-393 and Windows* logo for proxy offload
- Advanced Features:
  - -40 to 85 °C industrial temperature
  - Audio-video bridging
    - IEEE 1588/802.1AS precision time synchronization
    - IEEE 802.1Qav traffic shaper (with software extensions)
  - Jumbo frames
  - Interrupt moderation, VLAN support, IP checksum offload
  - Four transmit and four receive queues
  - RSS and MSI-X to lower CPU utilization in multi-core systems
  - Advanced cable diagnostics, auto MDI-X
  - Error Correcting Memory (ECC) in packet buffers
  - Four Software Definable Pins (SDPs)

January 2021
Revision Number: 1.8
Order No. 335761-009

Revision History

Rev	Date	Notes
1.8	January 2021	Added Section 1.4.3 (ROM-based Firmware). Updated Section 1.4.4 [Internal Non-Volatile Memory (iNVM)].
1.7	October 2018	I210-CL: AEC-Q100 Rev H certified to Features list on title page.
1.6	June 2018	Removed (available Q2’18) from the I210-CL: AEC-Q100 grade 3 certified feature on title page.
1.5	June 2018	Updated Section 6.2.2.2.3 [LaunchTime (25)].
1.4	April 2018	Updated Features table (I210-CL: AEC-Q100 grade 3 certified (-40 °C to 85 °C) with less than or equal to 20 DPM (available Q2'18).
1.3	March 2018	Updated Features table (I210-CL: AEC-Q100 grade 3 certified (-40 °C to 85 °C) with less than or equal to 100 DPM (available Q2'18). Changed Atmel* to Adesto* for Flash devices. Updated section 3.3.2.4 (Version Information).
1.2	January 2018	Updated Table 2-1 (Pull-Up/Pull-Down Resistors). Added section 3.3.2.4 (iNVM Structure Version Information). Updated section 5.5.6 (Timing Guarantees). Updated section 9.8.1 (Flash Devices).
1.1	June 2017	Updated section 2.0 (Pin Interface, changed pins 2-3, 5-9, 34-36, and 43-44 to RSVD).
1.0	May 2017	Initial release (Intel Public).

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



1.0 Introduction

The Intel® Ethernet Controller I210-CS/CL (I210-CS/CL) is a single port, compact, low power component that supports GbE designs. The I210-CS/CL offers a fully-integrated GbE Media Access Control (MAC) and a SGMII/SerDes port that can be connected to an external PHY. The I210-CS/CL supports PCI Express* [PCIe v2.1 (2.5GT/s)].

1.1 Scope

This document provides the external architecture (including device operation, pin descriptions, register definitions, etc.) for the I210-CS/CL.

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

1.2 Terminology and Acronyms

Table 1-1. Glossary

Definition	Meaning
1000BASE-BX	1000BASE-BX is the PICMG 3.1 electrical specification for transmitting 1 Gb/s Ethernet or 1 Gb/s fibre channel encoded data over the backplane.
1000BASE-KX	1000BASE-KX is the IEEE802.3ap electrical specification for transmitting 1 Gb/s Ethernet over the backplane.
1000BASE-CX	1000BASE-X over specialty shielded 150 Ω balanced copper jumper cable assemblies as specified in IEEE 802.3 Clause 39.
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
b/w	Bandwidth.
BIOS	Basic Input/Output System.
BMC	Baseboard Management Controller - often used interchangeably with Manageability Controller (MC).
BT	Bit time.
CRC	Cyclic redundancy check
DCA	Direct Cache Access.
DDOFF	Dynamic Device Off
DFT	Design for Testability.
DQ	Descriptor Queue.
DMTF	Distributed Management Task Force standard body.
DW	Double word (4 bytes).
EEE	Energy Efficient Ethernet - IEEE802.3az standard

Table 1-1. Glossary (Continued)

Definition	Meaning
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.
HPC	High - Performance Computing.
IPC	Inter Processor Communication.
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.
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).
TPH	TLP Process Hints (PCIe protocol).
PCS	Physical Coding Sub layer.
PHY	Physical Layer Device.
PMA	Physical Medium Attachment.
PMD	Physical Medium Dependent.
SA	Source Address.
SDP	Software Defined Pins.

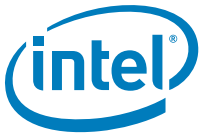


Table 1-1. Glossary (Continued)

Definition	Meaning
SerDes	Serializer/deserializer. A transceiver that converts parallel data to serial data and vice-versa.
SFD	Start Frame Delimiter.
SGMII	Serialized Gigabit Media Independent Interface.
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.
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).

1.2.1 External Specification and Documents

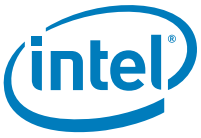
The I210-CS/CL implements features from the following specifications.

1.2.1.1 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 standard 802.1Q for VLAN
4. PICMG3.1 Ethernet/Fibre Channel Over PICMG 3.0 Draft Specification, January 14, 2003, Version D1.0
5. Serial-GMII Specification, Cisco Systems document ENG-46158, Revision 1.7
6. INF-8074i Specification for SFP (Small Form factor Pluggable) Transceiver (ftp://ftp.seagate.com/sff)
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 Draft 2.0, February 22, 2008
10. IEEE 802.1BF Ethernet Support for the IEEE P802.1AS Time Synchronization Protocol Task Force
11. 802.1BA - Audio Video Bridging (AVB) Systems
12. 802.1Qav - Forwarding and Queuing Enhancements for Time-Sensitive Streams

1.2.1.2 Host Interface Documents

1. PCI-Express 2.1 Base specification
2. PCI Specification, version 3.0
3. PCI Bus Power Management Interface Specification, Rev. 1.2, March 2004
4. Advanced Configuration and Power Interface Specification, Rev 2.0b, October 2002



1.2.1.3 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.

1.2.1.4 Proxy Documents

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

1.3 Product Overview

The I210-CS/CL supports a SerDes/SGMII MAC-to-MAC connection across a backplane a MAC-to-switch connection or a MAC-to-external PHY connection to communicate with automotive devices.

1.3.1 Audio/Video Bridging Support

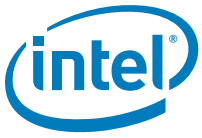
The I210-CS/CL supports IEEE 802.1 Audio Video Bridging (AVB) specifications. The draft AVB standards are designed to work over widely-used IEEE 802 layer 2 networks. These new standards provide networking features for tightly controlled media stream synchronization, buffering and reservation. The IEEE 802.1AVB task group is working on an interoperability standards for systems based on the AVB document set. This provides a complete list of parameters to plug into the various standards that are needed to build an AVB system. A simple grid of features versus market has been created as a first cut to describe four proposed interoperability profiles. Those profiles include Consumer Electronics, Professional A/V, Industrial, and Automotive. Use of AVB enables higher layer protocols and applications to realize professional-quality A/V even if there are various lower-layer network links in the path between endpoint devices.

The I210-CS/CL implements 4 receive queues and 4 transmit queues, where up to two queues are dedicated for stream reservation or priority, and up to three queues for strict priority. In Qav mode, the MAC flow control is disabled. Note that Qav mode is supported only in 100 Mb/s and 1000 Mb/s. Furthermore, Qav is supported only in full-duplex mode with no option for Jumbo packets transmission.

1.4 External Interface

1.4.1 PCIe Interface

The PCIe v2.1 (2.5GT/s) Interface is used by the I210-CS/CL as a host interface. The interface only supports the PCIe v2.1 (2.5GT/s) rate and is configured to x1. The maximum aggregated raw bandwidth for a typical PCIe v2.1 (2.5GT/s) configuration is 4 Gb/s in each direction. Refer to Section 2.3.1 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 v2.1 (2.5GT/s) specification.



1.4.2 Network Interfaces

Two interface types are used to connect the I210-CS/CL port to external devices. The following protocols are supported:

- SerDes interface to connect over a backplane to another SerDes compliant device or to an optical module. The I210-CS/CL supports both 1000BASE-BX and 1000BASE-KX (Without IEEE802.3ap backplane auto-negotiation)
- SGMII interface to attach to an external PHY, either on board or via an SFP module. The SGMII interface shares the same pins as the SerDes interface.

Refer to for a full pin description. For additional interface details, refer to Section 9.6.3 and Section 9.6.4.

1.4.3 ROM-based Firmware

Immutable firmware code is built into the ROM inside the I210-CS/CL to eliminate firmware tampering or firmware replacement.

1.4.4 Internal Non-Volatile Memory (iNVM)

Note: For security reasons, the I210-CS/CL has a lock-out mechanism after the iNVM is programmed at this stage, to prevent any tampering/retry of the iNVM programming.

The I210-CS/CL stores product configuration information in an Internal Non-Volatile Memory (iNVM) and in Flash memory. The I210-CS/CL does not support an external EEPROM. The I210-CS/CL supports a Flash-less mode where all the setup found normally in Flash memory are either set to their default, configured by software, or stored into memory.

Note: When operated in Flash-less mode with an external PHY (such as the I210-CS/CL SGMII SKU), no link up is made possible after power up before the driver configures the external PHY. It means that in such case, WoL is not supported when the system passes through the following states: G3 --> S5 --> WoL.

1.4.5 Serial Flash Interface

The I210-CS/CL provides an external SPI serial interface to a Flash for storing product configuration information and a boot ROM device such as the Winbond W25X80-BVSNIG or compatible Flash device. Refer to Section 9.8.1 for a list of validated or compatible Flash devices. The I210-CS/CL supports serial Flash devices with up to 64 Mb (8 MB) of memory. The size of the Flash used by the I210-CS/CL can be configured by the Flash itself. Refer to Section 2.3.2 for full pin description and Section 9.6.2.3 for timing characteristics of this interface.

Note: Though the I210-CS/CL supports devices with up to 8 MB of memory, bigger devices can also be used. Accesses to memory beyond the Flash device size results in access wrapping as only the lower address bits are used by the Flash device.

The I210-CS/CL can be used to connect an external Flash only for development purposes (such as speeding up debugging of prototypes). An external Flash is not supported for this production.

1.4.6 MDIO/I^{2}C 2-Wire Interface

The I210-CS/CL implements a management Interface to control an optional external PHY. The interface can be either a 2-wire Standard-mode I^{2}C interface used to control an SFP module or an MII Management Interface (also known as the Management Data Input/Output or MDIO Interface) for control plane connection between the MAC and PHY devices (master side). This interface provides the MAC and software with the ability to monitor and control the state of the external PHY. The I210-CS/CL supports the data formats defined in IEEE 802.3 clause 22.

Refer to Section 2.3.5 for a full pin description, Section 9.6.2.4 for MDIO timing characteristics, and Section 9.6.2.2 for I^{2}C timing characteristics of this interface.

The I^{2}C interface can alternatively be run over the SDP 0 and SDP2 pins. This can be useful when the I210-CS/CL operates with a copper PHY since the dedicated SFPx_I2C pins are not available in this mode for the control of other external devices.

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

The I210-CS/CL 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(refer to Section 6.2.21, Section 7.2.1 and Section 7.2.3), 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 6.8.3.3. Refer to Section 2.3.6 for pin description of this interface.

1.4.8 LED Interface

The I210-CS/CL 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 fields (refer to Section 6.2.18 and Section 6.2.20), thereby supporting LED displays configurable to a particular OEM preference.

Refer to Section 2.3.5 for full pin description of this interface.

Refer to Section 6.5 for more detailed description of LED behavior.

Table 1-2 to Table 1-9 list the I210 and I211 features.

Table 1-2. I210-CS/CL Features

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
Number of ports	1	1	1
Serial Flash interface	Y[1]	N	(Testing Only)
Integrated NVM (iNVM)	Y[2]	Y	Y[2]
4-wire SPI EEPROM interface	N	N	N
Configurable LED operation for software or OEM custom-tailoring of LED displays	Y	Y	Y
Protected Flash space for private configuration	Y[1]	N	N

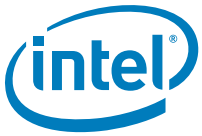


Table 1-2. I210-CS/CL Features (Continued)

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
Device disable capability	Y	Y	Y
Package size (mm x mm)	9x9	9x9	9x9
Embedded thermal sensor	N	N	N
Embedded thermal diode	N	N	N
Watchdog timer	Y	Y	Y
Boundary-Scan IEEE 1149.1	Y	Y	Y
Boundary-Scan IEEE 1149.6	N	N	N
Industrial temp (special SKU)	Y	N	Y

1. Not applicable in Flash-less I210 operation.
2. Flash-less I210 operation is supported (with no support for manageability related functionalities). Refer to the note that describes the limitation in Section 1.4.3.

Table 1-3. Network Features

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
Half duplex at 10/100 Mb/s operation	Y	Y	Full duplex operation at all supported speeds
10/100/1000 copper PHY integrated on-chip	1 port	1 port	N
Jumbo frames supported	Y	Y	Y
Size of jumbo frames supported	9.5 KB	9.5 KB	9018 bytes
Flow control support: send/receive PAUSE frames and receive FIFO thresholds	Y	Y	Y
Statistics for management and RMON	Y	Y	Y
802.1q VLAN support	Y	Y	Y
802.3az EEE support	Y	Y	N
MDI flip	N	N	N
SerDes interface for external PHY connection or system interconnect	Y	N	Y
1000BASE-KX interface for blade server backplane connections	Y	N	Y
802.3ap backplane auto-negotiation	N	N	N
SGMII interface for external 1000BASE-T PHY connection	1 port	N	1 port
SerDes support of non-auto-negotiation partner	Y	N	Y
SerDes signal detect	Y	N	Y
External PHY control I/F MDC/MDIO 2-wire I/F	Y	N	Y

Table 1-4. Host Interface Features

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
PCIe revision	2.1	2.1	2.1
PCIe physical layer	Gen 1	Gen 1	Gen 1
Bus width	x1	x1	x1

Table 1-4. Host Interface Features (Continued)

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
64-bit address support for systems using more than 4 GB of physical memory	Y	Y	Y
Outstanding requests for Tx buffers per port	6	6	6
Outstanding requests for Tx descriptors per port	1	1	1
Outstanding requests for Rx descriptors per port	1	1	1
Credits for posted writes	4	4	4
Max payload size supported	512 bytes	512 bytes	512 bytes
Max request size supported	2 KB	2 KB	2 KB
Link layer retry buffer size	3.2 KB	3.2 KB	3.2 KB
Vital Product Data (VPD)	Y[1]	N	N
VPD size	1024B[1]	N/A	N/A
End to End CRC (ECRC)	Y	Y	Y
OBFF (Optimized Buffer Flush/Fill)	Y[2]	N	N
Latency Tolerance Reporting (LTR)	Y[2]	N	N
TPH	Y	Y	Y
CSR access via Configuration space	Y	Y	Y
Access Control Services (ACS)	N	N	N
Audio Video Bridging (AVB) support	Y	N	Y

1. Not supported in Flash-less I210 operation.
2. Disabled by default via Flash.

Table 1-5. LAN Functions Features

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
Programmable host memory receive buffers	Y	Y	Y
Descriptor ring management hardware for transmit and receive	Y	Y	Y
ACPI register set and power down functionality supporting D0 and D3 states	Y	Y	Y
Software controlled global reset bit (resets everything except the configuration registers)	Y	Y	Y
Software Definable Pins (SDPs) - per port	4	4	4
Four SDP pins can be configured as general purpose interrupts	Y	Y	N
Wake up	Y	Y	Y
Flexible wake-up filters	8	8	8
Flexible filters for queue assignment in normal operation	8	8	8
IPv6 wake-up filters	Y	Y	Y
Default configuration by the for all LEDs for pre-driver functionality	3 LEDs	3 LEDs	3 LEDs
LAN function disable capability	Y	Y	Y
Programmable memory transmit buffers	Y	Y	Y
Double VLAN	Y	Y	Y

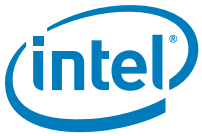


Table 1-5. LAN Functions Features (Continued)

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
IEEE 1588	Y	Y	Y
Per-packet timestamp	Y	Y	Y
Tx rate limiting per queue	Y	N	Y

Table 1-6. LAN Performance Features

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
TCP segmentation offload Up to 256 KB	Y	Y	Y
iSCSI TCP segmentation offload (CRC)	N	N	N
IPv6 support for IP/TCP and IP/UDP receive checksum offload	Y	Y	Y
Fragmented UDP checksum offload for packet reassembly	Y	Y	Y
Message Signaled Interrupts (MSI)	Y	Y	Y
Message Signaled Interrupts (MSI-X) number of vectors	5	5	5
Packet interrupt coalescing timers (packet timers) and absolute-delay interrupt timers for both transmit and receive operation	Y	Y	Y
Interrupt throttling control to limit maximum interrupt rate and improve CPU utilization	Y	Y	Y
Rx packet split header	Y	Y	Y
Receive Side Scaling (RSS) number of queues per port	Up to 4	Up to 2	Up to 4
Total number of Rx queues per port	4	2	4
Total number of TX queues per port	4	2	4
RX header replication Low latency interrupt DCA support TCP timer interrupts No snoop Relax ordering	Yes to all	Yes to all	Yes to all
TSO interleaving for reduced latency	Y	Y	Y
Receive Side Coalescing (RSC)	N	N	N
SCTP receive and transmit checksum offload	Y	Y	Y
UDP TSO	Y	Y	Y
IPSec offload	N	N	N

Table 1-7. Virtualization Related Features

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
Support for Virtual Machines Device queues (VMDq) per port	N	N	N
L2 MAC address filters (unicast and multicast)	16	16	16
L2 VLAN filters	Per port	Per port	Per port

Table 1-7. Virtualization Related Features (Continued)

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
PCI-SIG SR-IOV	N	N	N
Multicast/broadcast packet replication	N	N	N
VM to VM packet forwarding (packet loopback)	N	N	N
RSS replication	N	N	N
Traffic shaping	N	N	N
MAC and VLAN anti-spoofing	N	N	N
Malicious driver detection	N	N	N
Per-pool statistics	Y	Y	Y
Per-pool off loads	Y	Y	Y
Per-pool jumbo support	Y	Y	Y
Mirroring rules	N	N	N
External switch VEPA support	N	N	N
External switch NIV (VNTAG) support	N	N	N
Promiscuous modes	VLAN, unicast multicast	VLAN, unicast multicast	unicast multicast

Table 1-8. Manageability Features

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
Advanced pass-through-compatible management packet transmit/receive support	Y[1]	N	N
Managed ports on SMBus interface to external MC	1[1]	N	N
Auto-ARP reply over SMBus	Y[1]	N	N
NC-SI Interface to an external MC	Y[1]	N	N
Standard DMTF NC-SI protocol support	Y[1]	N	N
DMTF MCTP protocol over SMBus	Y[1]	N	N
NC-SI hardware arbitration	Y[1]	N	N
DMTF MCTP protocol over PCIe	Y[1]	N	N
Manageability L2 address filters	2	N	N
Manageability VLAN L2 filters	8	N	N
Manageability EtherType filters	4	N	N
Manageability Flex L4 port filters	8	N	N
Manageability Flex TCO filters	1	N	N
Manageability L3 address filters (IPv4)	4	N	N
Manageability L3 address filters (IPv6)	4	N	N
Proxying[2]	1 ARP Offload 2 NS Offloads MLD support mDNS[1]	1 ARP Offload 2 NS Offloads MLD support	1 ARP Offload 2 NS Offloads MLD support

1. is not supported in Flash-less I210 operation.
2. In Flash-less I210 operation, proxying support requires a dedicated firmware code be loaded to the device via the host interface (see Section 3.3.6).

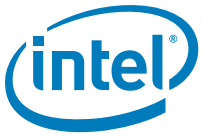


Table 1-9. Power Management Features

Feature	I210-AT/IT/IS (With Flash)	I211-AT	I210-CS/CL (iNVM Only)
Magic packet wake-up enable with unique MAC address	Y	Y	Y
ACPI register set and power down functionality supporting D0 and D3 states	Y	Y	Y
Full wake-up support (APM and ACPI 2.0)	Y	Y	Y
Smart power down at S0 no link and Sx no link	Y	Y	Y
LAN disable functionality (equivalent to Static device off functionality in the I210/I211)	Y[1]	Y[1]	Y
PCIe function disable	Y	Y	Y
Dynamic device off	Y[2]	Y[2]	Y
EEE	Y	Y	N
DMA coalescing	Y	N	N
OBFF/PE_WAKE_N	Y[3]	N	N

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. Disabled by default in Flash due to the lack of OBFF enabled platforms at initial release.

1.5 Overview of Changes Compared to the Intel® Ethernet Controller I350

The following section describes the modifications designed in the I210-CS/CL compared to the I350.

1.5.1 Network Interface

1.5.2 Audio and Video Bridging Support

See Section 1.3.1 for details on IEEE 802.1Qav support.

1.5.2.1 Tx Timestamp

The I210-CS/CL supports three types of transmit timestamps:

1. Reporting back of the timestamp in the transmit descriptor.
2. Inserting the timestamp in the packet sent.
3. Recording the timestamp of selected packet in a register (legacy behavior).

Transmit timestamp is described in Section 6.0.

1.5.3 Virtualization

SR-IOV and VMDq is not supported in hardware by the I210-CS/CL. The I210-CS/CL can still be used in virtualized systems where the VM switching is done in software.

1.5.3.1 Number of Exact Match Filters

The number of RAH/RAL registers is 16.

1.5.4 Host Interface

1.5.4.1 MSI-X Support

The number of MSI-X vectors supported by the I210-CS/CL changed to 5. For further information, refer to Section 6.3.

1.5.5 BOM Cost Reduction

1.5.5.1 On-chip 0.9V SVR Control

The I210-CS/CL includes a fully integrated on-chip Switching Voltage Regulator (SVR) that can be used to generate a 0.9V power supply without the need for a higher cost on-board 0.9V voltage regulator (refer to Section 3.5).

1.6 Device Data Flows

1.6.1 Transmit Data Flow

Table 1-10 lists a high level description of all data/control transformation steps needed for sending Ethernet packets to the line.

Table 1-10. Transmit Data Flow

Step	Description
1	The host creates a descriptor ring and configures one of the I210-CS/CL'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	The I210-CS/CL'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.
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.

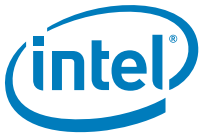


Table 1-10. Transmit Data Flow (Continued)

Step	Description
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 the I210-CS/CL and the driver can release the buffers.

1.6.2 Receive Data Flow

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

Table 1-11. Receive Data Flow

Step	Description
1	The host creates a descriptor ring and configures one of the I210-CS/CL'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	The I210-CS/CL'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, the I210-CS/CL 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.

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

2.0 Pin Interface

2.1 Pin Assignments

The I210-CS/CL supports a 64-pin, 9 x 9 QFN package with an Exposed Pad* (e-Pad*). Note that the e-Pad is ground.

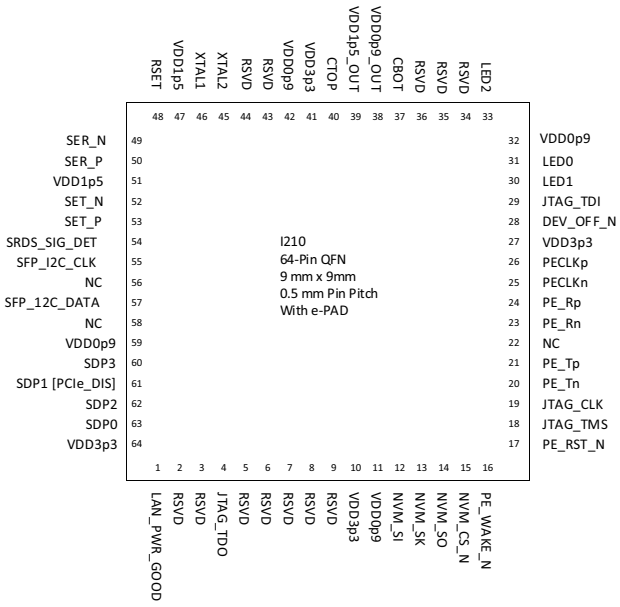


Figure 2-1. I210-CS/CL 64-Pin, 9 x 9 QFN Package With e-Pad

2.2 Pull-Up/Pull-Down Resistors

Table 2-1 lists internal and external pull-up/pull-down resistors and their functionality in different device states.

- As stated in the name and function table columns, the internal Pull-Up/Pull-Down (PU/PD) resistor values are 30 KΩ ± 50%.
- Only relevant (digital) pins are listed; analog or bias and power pins have specific considerations listed in Chapter 9.0.

Note: Refer to Section 10.0 for a list of board design schematic checklists, layout checklists, and reference design schematics for more details.

The device states are defined as follows:

- Power-up = while 3.3V is stable, yet 1.0V isn't
- Active = normal mode (not power up or disable)
- Disable = device off or dynamic device off – refer to Section 4.4

Table 2-1. Pull-Up/Pull-Down Resistors

Signal Name	Power Up[1]		Active		Disable[2]		External
	PU	Comments	PU	Comments	PU	Comments	
LAN_PWR_GOOD	N		N		N		Y
PE_WAKE_N	N		N		N		Y
PE_RST_N	N		N		N		PU[3]
NVM_SI	N		N		Y		PD/PU[4]
NVM_SO	Y		Y		Y		N
NVM_SK	Y		N		Y		N
NVM_CS_N	Y		N		Y		N
SMBD	N		N		N		Y
SMBCLK	N		N		N		Y
SMBALRT_N	N		N		N		Y
NCSI_CLK_IN	N	HiZ	N		N		PD
NCSI_CRS_DV	N	HiZ	N		N		PD
NCSI_RXD[1:0]	N	HiZ	N		N		PU
NCSI_TX_EN	N	HiZ	N		N		PD
NCSI_TXD[1:0]	N	HiZ	N		N		PU
SDP0	Y		Y	Until Flash auto-load done	Y	Might keep state by Flash control	N
SDP1	Y		Y	Until Flash auto-load done	Y	Might keep state by Flash control	N
SDP2	Y		Y	Until Flash auto-load done	Y	Might keep state by Flash control	N

Table 2-1. Pull-Up/Pull-Down Resistors (Continued)

Signal Name	Power Up[1]		Active		Disable[2]		External
	PU	Comments	PU	Comments	PU	Comments	
SDP3	Y		Y	Until Flash auto-load done	Y	Might keep state by Flash control	N
DEV_OFF_N	Y		N		N		PU optional if NC-SI is not used.
SRDS_SIG_DET	Y		N		N		Must be connected on board
SFP_I2C_CLK	Y		Y	Until Flash auto-load done or if *I2C disable* set in Flash	Y		Y if I2C
SFP_I2C_DATA	Y		Y	Until Flash auto-load done or if *I2C disable* set in Flash	Y		Y
LED0	Y		N		N	HiZ	
LED1	Y		N		N	HiZ	
LED2	Y		N		N	HiZ	
JTAG_CLK	N		N		N		Y[5]
JTAG_TDI	N		N		N		Y
JTAG_TDO	N		N		N		Y[5]
JTAG_TMS	N		N		N		Y[5]

1. Power up - LAN_PWR_GOOD = 0b
2. Refer to Section 5.2.6 for description of disable state.
3. 10 KΩ to 100 KΩ pull up can be used.
4. When pulled up, Flash security features are enabled.
5. These pins can be pulled up or pulled down (design dependent) when no clock device is connected to it.

2.3 Signal Type Definition

In	Input is a standard input-only signal.
Out (O)	Totem pole output is a standard active driver.
T/s	Tri-State is a bi-directional, tri-state input/output pin.
S/t/s	Sustained tri-state is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives an s/t/s pin low must drive it high for at least one clock before letting it float. A new agent cannot start driving an s/t/s signal any sooner than one clock after the previous owner tri-states it.
O/d	Open drain enables multiple devices to share as a wire-OR.
A-in	Analog input signals.
A-out	Analog output signals.
B	Input bias.
NCSI_in	NCSI input signal.
NCSI-out	NCSI output signal.

2.3.1 PCIe

Table 2-2. PCIe

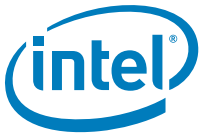
Symbol	Lead #	Type	Op Mode	Name and Function
PECLKp PECLKn	26 25	A-in	Input	PCIe Differential Reference Clock In This pin receives a 100 MHz differential clock input. This clock is used as the reference clock for the PCIe Tx/Rx circuitry and by the PCIe core PLL to generate a 125 MHz clock and 250 MHz clock for the PCIe core logic.
PE_Tp PE_Tn	21 20	A-out	Output	PCIe Serial Data Output Serial differential output link in the PCIe interface running at 2.5 Gb/s. This output carries both data and an embedded 2.5 GHz clock that is recovered along with data at the receiving end.
PE_Rp PE_Rn	24 23	A-in	Input	PCIe Serial Data Input Serial differential input link in the PCIe interface running at 2.5 Gb/s. The embedded clock present in this input is recovered along with the data.
PE_WAKE_N	16	T/s	Bi-dir	Wake The I210-CS/CL drives this signal to zero when it detects a wake-up event and either: • The PME_en bit in PMCSR is 1b or • The APME bit of the Wake Up Control (WUC) register is 1b.
PE_RST_N	17	In	Input	Power and Clock Good Indication The PE_RST_N signal indicates that both PCIe power and clock are available.

2.3.2 Flash

Note: These pins are used for testing only.

Table 2-3. Flash

Symbol	Lead #	Type	Op Mode	Name and Function
NVM_SI	12	T/s	Output	Serial Data Output Connect this lead to the input of the Flash.
NVM_SO	14	T/s	Input	Serial Data Input Connect this lead to the output of the Flash.
NVM_SK	13	T/s	Output	Non-Volatile Memory Serial Clock
NVM_CS_N	15	T/s	Output	Non-Volatile Memory Chip Select Output



2.3.3 Testability

Table 2-4. Testability

Symbol	Lead #	Type	Op Mode	Name and Function
JTAG_TDI	29	In	Input	JTAG TDI Input.
JTAG_CLK	19	In	Input	JTAG Clock Input.
JTAG_TMS	18	In	Input	JTAG Test Mode Select. This input controls the transitions of the test interface state machine.
JTAG_TDO	4	O/D		JTAG TDO

2.3.4 LEDs

Table 2-5 lists the functionality of each LED output pin. The default activity of each LED can be modified in the Flash. The LED functionality is reflected and can be further modified in the configuration registers (LEDCTL).

Table 2-5. LEDs

Symbol	Lead #	Type	Op Mode	Name and Function
LED0	31	Out	Output	Programmable LED number 0.
LED1	30	Out	Output	Programmable LED number 1.
LED2	33	Out	Output	Programmable LED number 2.

2.3.5 PHY Pins

Note: The I210-CS/CL has built in termination resistors. As a result, external termination resistors should not be used.

Table 2-6. PHY Pins

Symbol	Lead #	Type	Op Mode	Name and Function
SFP_I2C_DATA	57	A	Bi-dir	In SerDes: SFP 2 wire interface data – connects to Mod-Def2 pin of SFP (O/D). Can also be used as MDIO pin (T/S).
SFP_I2C_CLK	55	A	Bi-dir	In SerDes: SFP 2 wire interface clock – connects to Mod-Def1 input of SFP (O/D). Can also be used as MDC pin (Out).
SRDS_SIG_DET	54	A	Bi-dir	In SerDes: Signal Detect: Indicates that signal (light) is detected from the fiber. High for signal detect, low otherwise. Polarity of Signal Detect pin is controlled by the CTRL.ILOS bit. For non-fiber SerDes applications, link indication is internal, CONNSW.ENRGSRC bit should be 0b and pin should be connected to a pull-up resistor.

Table 2-6. PHY Pins

Symbol	Lead #	Type	Op Mode	Name and Function
SET_P SET_N SET_P SER_N	53 52 50 49	A	Bi-dir	In SerDes SerDes/SGMII Serial Data input/output: Differential SERDES Receive/Transmit interface. A serial differential input/output pair running at 1.25Gb/s. An embedded clock present in this input is recovered along with the data. This output carries both data and an embedded 1.25 GHz clock that is recovered along with data at the receiving end.
XTAL1 XTAL2	46 45	A-In A-Out	Input/ Output	XTAL In/Out These pins can be driven by an external 25 MHz crystal or driven by an external MOS level 25 MHz oscillator. Used to drive the PHY.
RSET	48	A	Bias	PHY Termination This pin should be connected through a 4.99 KΩ ±1% resister to ground.

2.3.6 Miscellaneous Pins

Table 2-7. Miscellaneous Pins

Symbol	Lead #	Type	Op Mode	Name and Function
DEV_OFF_N	28	In	Input	This is a 3.3V input signal. Asserting DEV_OFF_N puts the I210-CS/CL in device disable mode. Note that this pin is asynchronous. Functionality of this input can be changed by Flash bits settings - see Table 2-9 for more details.
SDP0	63	T/s	Input/ Output	Software defined pin 0.
SDP1 [PCIe_DIS]SDP1	61	T/s	Input/ Output	Software defined pin 1. See Table 2-9 for PCIe function disable settings.
SDP2	62	T/s	Input/ Output	Software defined pin 2.
SDP3	60	T/s	Input/ Output	Software defined pin 3.
LAN_PWR_GOOD	1	In	Input	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 VDDO.
NC	22	Voltage	Input	Optional pin used to connect an external power supply to the PCIe block in order to replace the internal LDO.

2.3.7 Power Supplies and Support Pins

2.3.7.1 Power Support

Table 2-8. Power Support

Symbol	Lead #	Type / Voltage	Name and Function
CBOT	37	A-in A-Out	Capacitor bottom connection.
CTOP	40	A-In capacitor A-Out	Capacitor top connection.

Note: These pins must be connected together by a 39 nF capacitor (refer to capacitor part # GRM155R61A393KA01).

2.3.7.2 Power Supply

2.4 Strapping Options

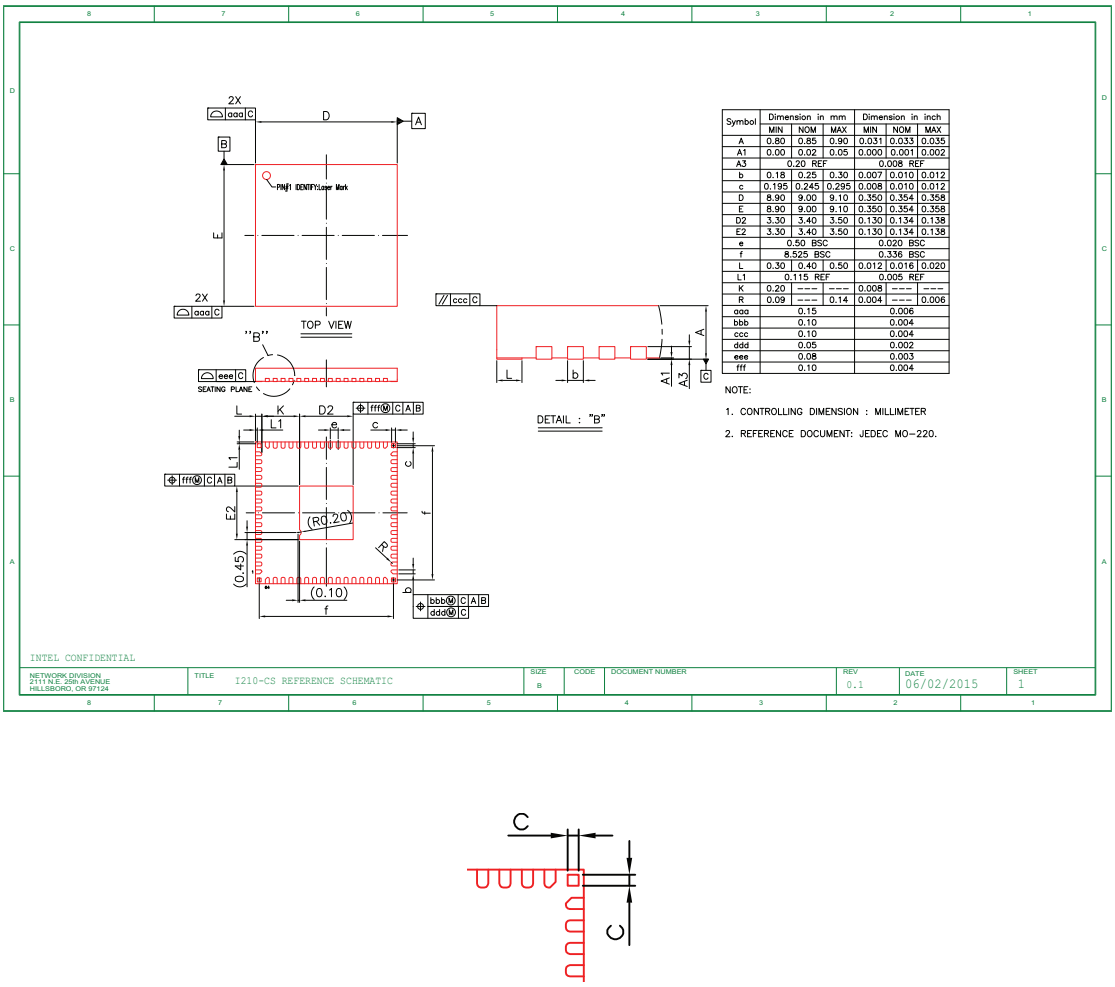
Table 2-9. Strapping Options

		Pad					NVM				PU	
							0x1E:15	0x29.10	0x29.13	0x29.15	Internal PU	
Function	**Latch Event**	DEV_OFF_N	SDP3	NVM_SK	NVM_SI	SDP1 [PCIe_DIS]	Device Off Enable	nvm_aux_pwr_en	nvm_alt_aux_pwr_en	en_pin_pcie_func_dis		**Comments**
DEV_OFF_N	N/A	0	X	X	X	X	1	X	X	X		Device off mode when the pin is pulled low.
AUX_PWR (option 1)	N/A	1	X	X	X	X	0	1	X	X		AUX power mode when the pin is pulled high.
AUX_PWR (option 2)	N/A	X	1	X	X	X	0	0	1	X		AUX power mode when the pin is pulled high.
SECURITY_EN	LAN_PWR_GOOD	X	X	X	1	X	X	X	X	X	PU (until LPG)	Flash security is disabled when the pin is pulled low.
PCIE_DIS_N	N/A	X	X	X	X	0	X	X	X	1		Active low, valid on Flash load complete. Strap logic that requires a dedicated SDP.

Note: nvm_aux_pwr_en and nvm_alt_aux_pwr_en bits are read as 0b from NVM, AUX_PWR mode is enabled.

2.5 Package

The I210-CS/CL supports a 64-pin, 9 x 9 QFN package with e-Pad. Figure 2-2 shows the package schematics.



Symbol	Dimension in mm			Dimension in inch		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.80	0.85	0.90	0.031	0.033	0.035
A1	0.00	0.02	0.05	0.000	0.001	0.002
A3		0.20 REF			0.008 REF	
b	0.18	0.25	0.30	0.007	0.010	0.012
c	0.195	0.245	0.295	0.008	0.010	0.012
D	8.90	9.00	9.10	0.350	0.354	0.358
E	8.90	9.00	9.10	0.350	0.354	0.358
D2	3.30	3.40	3.50	0.130	0.134	0.138
E2	3.30	3.40	3.50	0.130	0.134	0.138
e		0.50 BSC			0.020 BSC	
f		8.525 BSC			0.336 BSC	
L	0.30	0.40	0.50	0.012	0.016	0.020
L1		0.115 REF			0.005 REF	
K	0.20	---	---	0.008	---	---
R	0.09	---	0.14	0.004	---	0.006
aaa		0.15			0.006	
bbb		0.10			0.004	
ccc		0.10			0.004	
ddd		0.05			0.002	
eee		0.08			0.003	
fff		0.10			0.004	

Figure 2-2. I210-CS/CL QFN 9 x 9 mm Package

3.0 Interconnects

3.1 PCIe

3.1.1 PCIe Overview

PCIe is a third generation I/O architecture that enables cost competitive next generation 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.

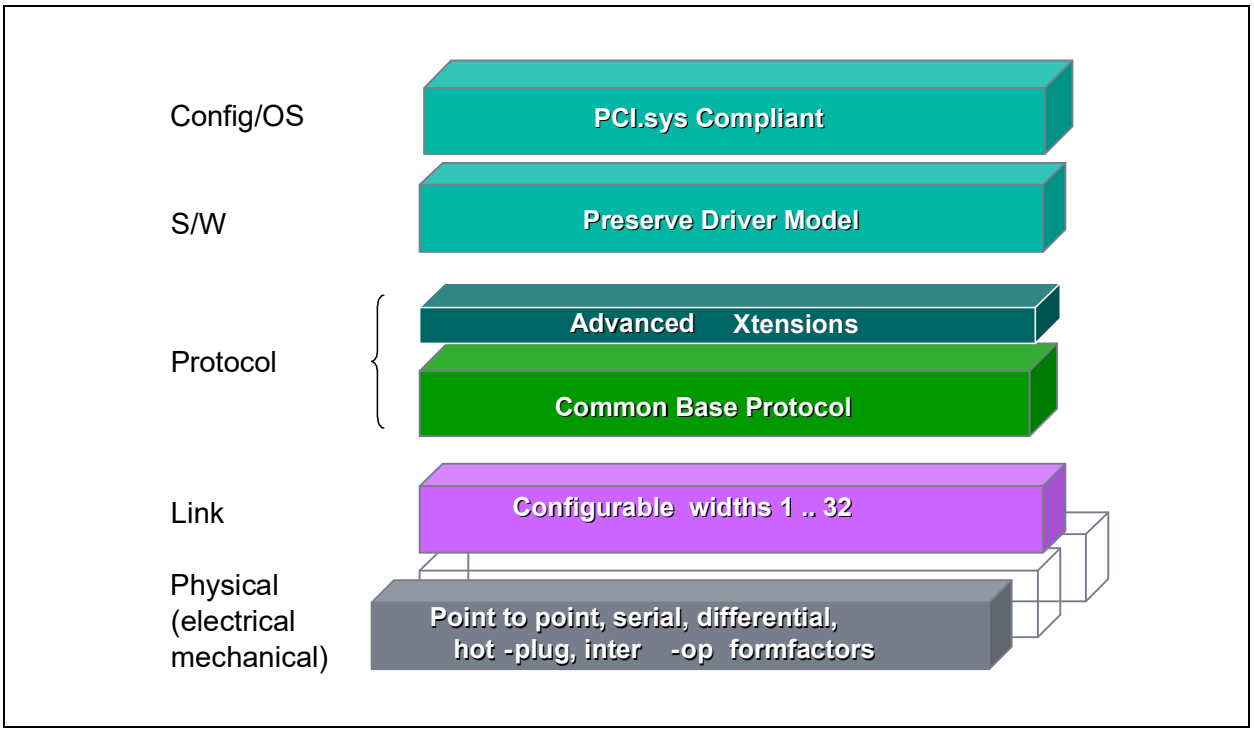


Figure 3-1. PCIe Stack Structure

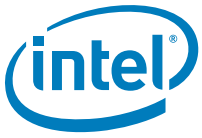
PCIe's physical layer consists of a differential transmit pair and a differential receive pair. Full-duplex data on these two point-to-point connections is self-c such that no dedicated clock signals are required. The bandwidth of this interface increases linearly with frequency.

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.

3.1.1.1 Architecture, Transaction and Link Layer Properties

- Split transaction, packet-based protocol
- Common flat address space for load/store access (such as PCI addressing model)
  - Memory address space of 32-bits to allow compact packet header (must be used to access addresses below 4 GB)
  - Memory address space of 64-bit using extended packet header
- Transaction layer mechanisms:
  - PCI-X style relaxed ordering
  - Optimizations for no-snoop transactions
- Credit-based flow control
- Packet sizes/formats:
  - Maximum upstream (write) payload size of 512 bytes
  - Maximum downstream (read) payload size of 512 bytes
- Reset/initialization:
  - Frequency/width/profile negotiation performed by hardware
- Data integrity support
  - Using CRC-32 for transaction layer packets
- Link layer retry for recovery following error detection
  - Using CRC-16 for link layer messages
- No retry following error detection
  - 8b/10b encoding with running disparity
- Software configuration mechanism:
  - Uses PCI configuration and bus enumeration model
  - PCIe-specific configuration registers mapped via PCI extended capability mechanism
- Baseline messaging:
  - In-band messaging of formerly side-band legacy signals (such as interrupts, etc.)
  - System-level power management supported via messages
- Power management:
  - Full support for PCI-PM
  - Wake capability from D3cold state
  - Compliant with ACPI, PCI-PM software model



  - — Active state power management
- Support for PCIe v2.1 (2.5GT/s)
  - — Support for completion time out
  - — Support for additional registers in the PCIe capability structure.

3.1.1.2 Physical Interface Properties

- Point to point interconnect
  - — Full-duplex; no arbitration
- Signaling technology:
  - — Low Voltage Differential (LVD)
  - — Embedded clock signaling using 8b/10b encoding scheme
- Serial frequency of operation: 2.5 Gb/s.
- Interface width of x1.
- DFT and DFM support for high volume manufacturing

3.1.1.3 Advanced Extensions

PCIe defines a set of optional features to enhance platform capabilities for specific usage modes. The I210-CS/CL supports the following optional features:

- Extended error reporting - messaging support to communicate multiple types/severity of errors.
- Device serial number.
- Completion timeout control.
- TLP Processing Hints (TPH) - provides hints on a per transaction basis to facilitate optimized processing of transactions that target memory space.

3.1.2 General Functionality

3.1.2.1 Native/Legacy

All the I210-CS/CL PCI functions are native PCIe functions.

3.1.2.2 Transactions

The I210-CS/CL 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 (the I210-CS/CL is a physical device). The I210-CS/CL implements a single logical device with one PCI function. The device number is captured from the 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	
0x6-017	Not used	
0x18	Descriptor Tx	
0x19-0x1B	Not used	
0x1C	Descriptor Rx	
0x1D-0x1F	Not used	

3.1.3.1.2 TAG ID Allocation for Write Transactions

Request tag allocation depends on these system parameters:

- DCA supported/not supported in the system (*DCA_CTRL.DCA_DIS* - refer to Section 7.13.4 for details)
- TPH enabled in the system.
- DCA enabled/disabled for each type of traffic (*TXCTL.TX Descriptor DCA EN, RXCTL.RX Descriptor DCA EN, RXCTL.RX Header DCA EN, RXCTL.Rx Payload DCA EN*).
- TPH enabled or disabled for the specific type of traffic carried by the TLP (*TXCTL.TX Descriptor TPH EN, RXCTL.RX Descriptor TPH EN, RXCTL.RX Header TPH EN, RXCTL.Rx Payload TPH EN*).
- System type: Legacy DCA vs. DCA 1.0 (*DCA_CTRL.DCA_MODE* - refer to Section 7.13.4 for details).
- CPU ID (*RXCTL.CPUID* or *TXCTL.CPUID*).

See the case studies below for information on different implementations

3.1.3.1.2.1 Case 1 - DCA Disabled in the System

Table 3-2 lists the write requests tags. Unlike read, the values are for debug only, allowing tracing of requests through the system.

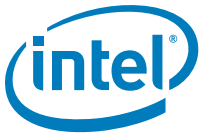


Table 3-2. IDs in Write Transactions (DCA Disabled Mode)

Tag ID	Description
0x0 - 0x1	Reserved
0x2	Tx descriptors write-back / Tx head write-back
0x3	Reserved
0x4	Rx descriptors write-back
0x5	Reserved
0x6	Write data
0x7 - 0x1D	Reserved
0x1E	MSI and MSI-X
0x1F	Reserved

3.1.3.1.2.2 Case 2 - DCA Enabled in the System, but Disabled for the Request

- Legacy DCA platforms - If DCA is disabled for the request, the tags allocation is identical to the case where DCA is disabled in the system. Refer to Table 3-2.
- DCA 1.0 platforms - All write requests have a tag value of 0x00.

Note: When in DCA 1.0 mode, messages and MSI/MSI-X write requests are sent with the no-hint tag.

3.1.3.1.2.3 Case 3 - DCA Enabled in the System, DCA Enabled for the Request

- Legacy DCA platforms: the request tag is constructed as follows:
  - Bit[0] – DCA Enable
  - Bits[3:1] - The *CPU ID* field taken from the CPUID[2:0] bits of the RXCTL or TXCTL registers
  - Bits[7:4] - Reserved
- DCA 1.0 platforms: the request tag (all 8 bits) is taken from the *CPUID* field of the RXCTL or TXCTL registers

3.1.3.1.2.4 Case 4 - TPH Enabled in the System, TPH Enabled for the Request

- The request tag (all 8 bits) is taken from the *CPUID* field of the adequate register or context as listed in Table 6-60.

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. The I210-CS/CL 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 8.4.5.11).

The I210-CS/CL's reaction in case of a completion timeout is listed in Table 3-12.

The I210-CS/CL 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 3-3.

Table 3-3. Completion Timeout Programming

Capability	Programming capability
Completion Timeout Enabling	Controlled through *PCI Device Control 2* configuration register.
Resend Request Enable	Loaded from the Flashinto 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* Flash 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

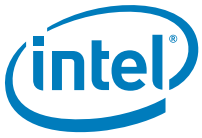
Programmed through the PCI Device Control 2 configuration register (refer to Section 8.4.5.11). The I210-CS/CL supports all ranges defined by PCIe v2.1 (2.5GT/s).

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 the I210-CS/CL is installed. This ensures that the I210-CS/CL 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 Transaction Layer

The upper layer of the PCIe architecture is the transaction layer. The transaction layer connects to the I210-CS/CL core using an implementation specific protocol. Through this core-to-transaction-layer protocol, the application-specific parts of the I210-CS/CL interact with the PCIe subsystem and transmit and receive requests to or from the remote PCIe agent, respectively.



3.1.4.1 Transaction Types Accepted by the I210-CS/CL

Table 3-4. 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

3.1.4.1.1 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 the I210-CS/CL where initialization is long due to the Flash read operation following reset.

If the read of the PCIe section in the Flash was not completed and the I210-CS/CL receives a configuration request, the I210-CS/CL 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.

3.1.4.1.2 Partial Memory Read and Write Requests

The I210-CS/CL 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.

3.1.4.2 Transaction Types Initiated by the I210-CS/CL

Table 3-5. 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 Flash (128 bytes, 256 bytes or 512 bytes). Effective MAX_PAYLOAD_SIZE is defined according to configuration space register.
2. MCTP messages contains payload.

3.1.4.2.1 Data Alignment

Requests must never specify an address/length combination that causes a memory space access to cross a 4 KB boundary. The I210-CS/CL 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 should 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 the I210-CS/CL's internal limitation.
3. A single request should not span across different memory pages as noted by the 4 KB boundary previously mentioned.

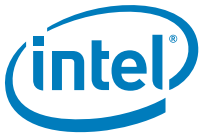
Note: The rules apply to all the I210-CS/CL requests (read/write, snoop and no snoop).

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 the I210-CS/CL 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.

3.1.4.2.2 Multiple Tx Data Read Requests (MULR)

The I210-CS/CL supports 6 pipelined requests for transmit data on the port. In general, the 6 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. The I210-CS/CL handles completions that arrive in any order. Once all completions arrive for a given request, the I210-CS/CL might issue the next pending read data request.

- The I210-CS/CL 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 6 pipeline requests for transmit data, the I210-CS/CL can issue up to one read request to fetch transmit descriptors and one 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.

3.1.4.3 Messages

3.1.4.3.1 Message Handling by the I210-CS/CL (as a Receiver)

Message packets are special packets that carry a message code.

The upstream device transmits special messages to the I210-CS/CL by using this mechanism.

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

Table 3-6. Supported Message in the I210-CS/CL (as a Receiver)

Message Code [7:0]	Routing r2r1r0	Message	I210-CS/CL 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
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 (see Section 3.1.4.4)	Send to MCTP reassembly if Vendor ID = 0x1AB4 (DMTF) and VDM code - 0000b (MCTP). Otherwise, silently drop

3.1.4.3.2 Message Handling by I210-CS/CL (as a Transmitter)

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

Table 3-7. Supported Message in the I210-CS/CL (as a Transmitter)

Message code [7:0]	Routing r2r1r0	Message
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
0x18	000	PM_PME
0x1B	101	PME_TO_ACK
0x10	100	Reserved
0x7F	000, 010, 011,	VDM (see Section 3.1.4.4)

3.1.4.4 Ordering Rules

The I210-CS/CL meets the PCIe ordering rules (PCI-X rules) by following the PCI simple device model:

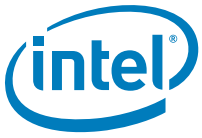
- Deadlock avoidance - Master and target accesses are independent. The response to a target access does not depend on the status of a master request to the bus. If master requests are blocked, such as due to no credits, target completions might still proceed (if credits are available).
- Descriptor/data ordering - The I210-CS/CL does not proceed with some internal actions until respective data writes have ended on the PCIe link:
  - The I210-CS/CL does not update an internal header pointer until the descriptors that the header pointer relates to are written to the PCIe link.
  - The I210-CS/CL does not issue a descriptor write until the data that the descriptor relates to is written to the PCIe link.

The I210-CS/CL might issue the following master read request from each of the following clients:

- One Rx Descriptor Read
- One Tx Descriptor Read
- Tx Data Read (up to 6)

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

3.1.4.4.1 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, the I210-CS/CL sorts the request completions and transfers them to the Ethernet in the correct order.

3.1.4.5 Transaction Definition and Attributes

3.1.4.5.1 Max Payload Size

The I210-CS/CL policy to determine Max Payload Size (MPS) is as follows:

- Master requests initiated by the I210-CS/CL (including completions) limits MPS to the value defined for the function issuing the request.
- Target write accesses to the I210-CS/CL 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.

3.1.4.5.2 Relaxed Ordering

The I210-CS/CL takes advantage of the relaxed ordering rules in PCIe. By setting the relaxed ordering bit in the packet header, the I210-CS/CL enables the system to optimize performance in the following cases:

- Relaxed ordering for descriptor and data reads: When the I210-CS/CL 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 the I210-CS/CL 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.
- The I210-CS/CL 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 the I210-CS/CL by clearing the *RO_DIS* bit in the CTRL_EXT register. Actual setting of relaxed ordering is done for LAN traffic by the host through the DCA registers.

3.1.4.5.3 Snoop Not Required

The I210-CS/CL 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 the I210-CS/CL by clearing the *NS_DIS* bit in the CTRL_EXT register. Actual setting of no snoop is done for LAN traffic by the host through the DCA registers.

3.1.4.5.4 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 3-8 lists software configuration for the *No-Snoop* and *Relaxed Ordering* bits for LAN traffic when I/OAT 2 is enabled.

Table 3-8. 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 3.1.4.5.4.1
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 3.1.4.5.4.1.

3.1.4.5.4.1 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 the I210-CS/CL should perform a no-snoop DMA transfer when it eventually writes a packet to this buffer.

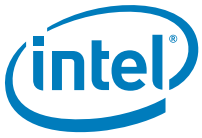
When a non-snoop transaction is activated, the TLP header has a non-snoop attribute in the *Transaction Descriptor* field.

This is triggered by the *NSE* bit in the receive descriptor. Refer to Section 6.1.4.2.

3.1.4.5.5 TLP Processing Hint (TPH)

The *TPH* bit can be set to provide information to the root complex about the cache in which the data should be stored or from which the data should be read as described in Section 6.7.2.

TPH is enabled via the *TPH Requester Enable* field in the TPH control register of the configuration space (refer to Section 8.5.3.3). Setting of the *TPH* bit for different type of traffic is listed in Table 6-60.



3.1.4.6 Flow Control

3.1.4.6.1 I210-CS/CL Flow Control Rules

The I210-CS/CL implements only the default Virtual Channel (VC0). A single set of credits is maintained for VC0.

Table 3-9. 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:

- The I210-CS/CL 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.
- The I210-CS/CL 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.
- The I210-CS/CL follows the PCIe recommendations for frequency of UpdateFC FCPs.

3.1.4.6.2 Upstream Flow Control Tracking

The I210-CS/CL 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).

3.1.4.6.3 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%).

3.1.4.6.4 Flow Control Timeout Mechanism

The I210-CS/CL 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).

3.1.4.7 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, the I210-CS/CL 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.

3.1.5 Data Link Layer

3.1.5.1 ACK/NAK Scheme

The I210-CS/CL 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

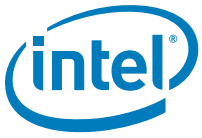
In all other cases, the I210-CS/CL schedules an ACK transmission according to time-outs specified in the PCIe specification (depends on link speed, link width, and max_payload_size).

3.1.5.2 Supported DLLPs

The following DLLPs are supported by the I210-CS/CL as a receiver:

Table 3-10. DLLPs Received by the I210-CS/CL

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 the I210-CS/CL as a transmitter:

Table 3-11. DLLPs Initiated by the I210-CS/CL

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.

3.1.5.3 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.

3.1.6 Physical Layer

3.1.6.1 Link Speed

- The I210-CS/CL supports only 2.5GT/s link speeds.

The I210-CS/CL 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.

The I210-CS/CL 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.

3.1.6.2 Link Width

The I210-CS/CL supports a maximum link width of x1.

During link configuration, the platform and the I210-CS/CL negotiate on a common link width. The link width must be x1.

3.1.6.3 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.

3.1.6.4 L0s Exit latency

The number of FTS sequences (N_FTS) sent during L1 exit, can be loaded from the Flash.

3.1.6.5 Reset

The PCIe PHY can supply a core reset to the I210-CS/CL. 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.

3.1.6.6 Scrambler Disable

The scrambler/de-scrambler functionality in the I210-CS/CL can be disabled by either one of the two connected devices according to the PCIe specification.

3.1.7 Error Events and Error Reporting

3.1.7.1 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 the I210-CS/CL.

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

3.1.7.2 Error Events

Table 3-12 lists the error events identified by the I210-CS/CL and the response in terms of logging, reporting, and actions taken. Consult the PCIe specification for the effect on the PCI Status register.

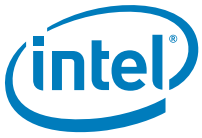


Table 3-12. Response and Reporting of PCIe Error Events

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			
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. The I210-CS/CL doesn't report violations of flow control initialization protocol

Table 3-12. 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	The I210-CS/CL doesn't check for this error and accepts these packets. This might cause a completion timeout condition.

3.1.7.3 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. The I210-CS/CL then reacts as listed in Table 3-12.

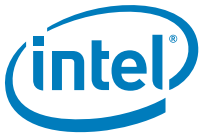
The I210-CS/CL does not initiate any additional master requests for that PCI function until it detects an internal software reset for the LAN port. Software is able to access device registers after such a fault.

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.

3.1.7.4 ECRC

The I210-CS/CL supports End to End CRC (ECRC) as defined in the PCIe specification. The following functionality is provided:

- Inserting an ECRC in all transmitted TLPs:
  - The I210-CS/CL indicates support for inserting ECRC in the *ECRC Generation Capable* bit of the PCIe configuration registers. This bit is loaded from the ECRC Generation Flash 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 Flash 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.
  - The I210-CS/CL indicates support for ECRC checking in the *ECRC Check Capable* bit of the PCIe configuration registers. This bit is loaded from the ECRC Check Flash bit.
  - ECRC checking is enabled by the *ECRC Check Enable* bit of the PCIe configuration registers.
- ECRC errors are reported.



3.1.7.5 Partial Read and Write Requests

3.1.7.5.1 Partial Memory Accesses

The I210-CS/CL 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.

The I210-CS/CL does not generate an error indication in response to any of the above events.

3.1.7.5.2 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

3.1.7.6 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.

3.1.7.7 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.

3.1.7.8 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 the I210-CS/CL. 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 the I210-CS/CL. 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 3.1.7.3.
- 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.

3.1.7.9 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.

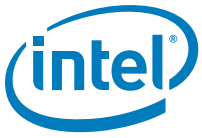
Note: Asserting *CTRL.DEV_RST* flushes any pending transactions on the PCIe and reset's the port.

3.1.8 PCIe Power Management

Described in Section 5.4.1 - Power Management.

3.1.9 PCIe Programming Interface

Described in Chapter 8.0 - PCIe Programming Interface



3.2 Flash

3.2.1 General Overview

The I210-CS/CL can use a Flash device for storing product configuration information. However, the purpose of this option is to make prototype debugging more convenient, it's not supported for production. Production designs should use an iNVM for their entire configuration. For more information about Flash, refer to the *Intel® Ethernet Controller I210 Datasheet*.

3.3 iNVM

The I210-CS/CL can operate with no Flash attached (Flash-less mode). The I210-CS/CL incorporates an on-die internal NVM (iNVM) memory (size 2 Kb) that enables designers to internally program the I210-CS/CL with a subset of the default values that are normally associated with an external Flash. iNVM is similar to One Time Programmable (OTP) memory except that it allows for a limited number of modifications and corrections after initial programming. The iNVM has a capacity of 64 words, or 32 two-word CSR entries. A word, once used, cannot be rewritten. The initial programming will take a number of these, and each new entry takes additional words until the capacity is reached. For example, programming the MAC address consumes three auto-load word structures.

The last two words of the iNVM (62-63) are used for manufacturing identification information. Word 61 is written with version information when the iNVM is programmed using Intel tools. The contents of these words do not affect the operation of the device.

Refer to the note in Section 1.4.4 for the functional limitation that exists when the I210-CS/CL operates without an external Flash part.

This section describes the iNVM structure for the I210-CS/CL.

3.3.1 iNVM Contents

The iNVM memory is used there to store and program the default values that are otherwise programmed via auto-load from the external Flash memory. The list of programmable values includes the following (amongst others):

- MAC address - words 0x00, 0x01, 0x02
  - Serial ID (for PCIe) is a derivative of MAC address.
- iNVM image revision - word 0x05
- Subsystem ID and Subsystem Vendor ID - words 0x0B, 0x0C
  - Needed only for NIC and for other vendors than Intel.
- Device ID - word 0x0D
  - Device ID: Use a separate device ID for the I211 running with a programmed iNVM (0x1539).
- Board Configuration (LEDs, SDPs, etc.) - words 0x1C, 0x1F, 0x20, 0x24
- LAN power consumption - word 0x22
- PHY/PCIe analog parameters. This information is loaded in the iNVM as it is determined based on the silicon’s process state.
  - Other critical PCIe settings that are loaded only at power-up.
- Hardware init, workaround/bypass
  - Initialization Control word 1 - word 0x0A to:

- set GPAR_EN bit to 1b (enable global parity check)
- optionally set iNVM to 1b (see note in Section 3.3.2.1)

— Initialization Control Word 2 to set TX_LPI_EN bit - word 0x0F

— Device Off Enable bit to 1b - word 0x1E

- PHY disconnect until the software device driver is up and running or WoL setup - words 0x24, 0x29
  — PHY disconnect is achieved by setting the *Go Link Disconnect* field (bit 5) in the PHPM register
- FLBAR_Size set to 0 - word 0x28

Not Supported in iNVM:

- Manageability and manageability parameters
- Pointers
- VPD
- Legacy Option ROM - PXE (PXE driver can reside in BIOS Flash), iSCSI boot (requires external Flash), etc.

3.3.2 iNVM Structures

The iNVM contains the following three structures:

1. Word auto-load (2 words)
2. CSR auto-load (4 words)
3. PHY register auto-load (2 words)

Each structure starts with a type field. When a non-null unknown type is encountered, the 32-bits are skipped by hardware as they might contain an item that is relevant to firmware or software.

When invalidating a structure, all its Type field bits should be set to 1b. That way, each time the device iNVM parser encounters a 111b type it can skip 32-bit words until the next non 111b type is detected.

Table 3-13 lists the different iNVM structure types:

Table 3-13. iNVM Structure Types

Type	Description
000b	Un-initialized iNVM Dword, stop iNVM parsing.
001b	Word auto-load
010b	CSR auto-load
011b	PHY register auto-load
100b	Reserved - do not use this value
111b	Invalidated iNVM structure, skip the Dword (16-bits)
Other	Reserved for future use, skip the Dword (16-bits)

Table 3-14 lists the iNVM value load condition as a function of the reset types. the reset type value should be the same as specified in the reset type value of the auto-load table.

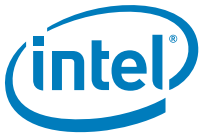


Table 3-14. iNVM Structure Reset Types (for Auto-load)

Reset Type	Description
00b	Load on Power-up (LAN_PWR_GOOD) reset
10b	Load on PCIe reset and power-up reset
01b	Reserved
11b	Load on software reset, PCIe reset, and power-up reset

3.3.2.1 Word Auto-load Structure

MSB[15] LSB[0]

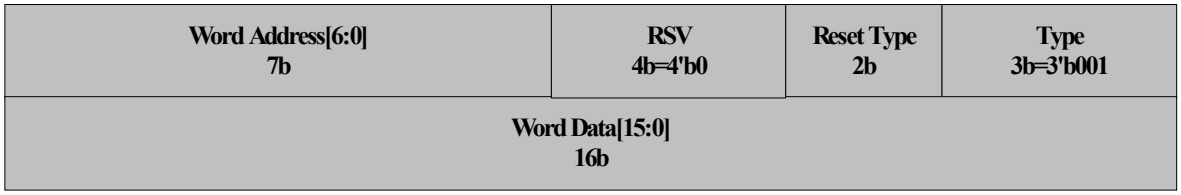


Figure 3-2. Word Auto-load Structure

3.3.2.1.1 iNVM Programmed Word Structures (Type 001b)

Table 3-15. iNVM Values

Word Address	Word Data (16 bits)
0x00	Ethernet Address Low
0x01	Ethernet Address Mid
0x02	Ethernet Address High
0x0A	Initialization Control Word 1 (Word 0x0A) - Section 6.2.2
0x0E	Vendor ID (Word 0x0E) - Section 6.2.6
0x0F	Initialization Control Word 2 (Offset 0x0F) - Section 6.2.7
0x1B	PCIe Control 1 (Word 0x1B) - Section 6.2.17
0x1C	LED1 Configuration Defaults (Word 0x1C) - Section 6.2.18
0x1E	Device Rev ID (Word 0x1E) - Section 6.2.19
0x1F	LED0,2 Configuration Defaults (Word 0x1F) - Section 6.2.20
0x20	Software Defined Pins Control (Word 0x20) - Section 6.2.21
0x21	Functions Control (Word 0x21) - Section 6.2.22
0x22	LAN Power Consumption (Word 0x22) - Section 6.2.23
0x24	Initialization Control 3 (Word 0x24) - Section 6.2.24
0x29	PCIe Control 3 (Word 0x29) - Section 6.2.26
0x28	PCIe Control 2 (Word 0x28) - Section 6.2.25
0x2E	Watchdog Configuration (Word 0x2E) - Section 6.2.29
0x31	Configuration Customization Options PCIe Function (Word 0x31) - Section 6.8.6.2

Note: In order to secure the iNVM memory (such as avoiding any further write to it after manufacturing), a word auto-load structure must be present in iNVM for setting the iNVM bit to 1b in word address 0x0A.

3.3.2.2 CSR Auto-load Structure

The CSR auto-load structure is defined as follows:

MSB[15] LSB[0]

RSV 11b=11'b0	Reset Type 2b	Type 3b=3'b010
CSR Address in DWord[15:0] (bit 15 is reserved) 16b		
CSR Data[15:0] 16b		
CSR Data[31:16] 16b		

Figure 3-3. CSR Auto-load Structure

3.3.2.3 PHY Register Auto-load Structure

MSB[15] LSB[0]

MDIC REGADD [4:0] 5b	RSV 6b=6'b0	Reset Type 2b=2'b11	Type 3b=3'b011
MDIC DATA[15:0] 16b			

Figure 3-4. PHY Register Auto-load Structure

Once the structure is loaded to the PHY, the EEMNGCTL.CFG_DONE bit is set.

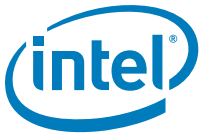
3.3.2.4 iNVM Structure (Version Information for Customer Use)

- Bit 2:0 = Invalidated
- bit 4:3 = Reserved 0x1
- bit 8:5 = Reserved 0xF
- Bit 31:16 = Version number (number of 1' s show the version number 0x1 = 1, 0x3 =2, 0x7 =3, etc.)

3.3.3 iNVM Programming Flows

iNVM can be programmed at several occasions and via different means:

1. At the chip manufacturing site (by Intel), via a special pin. It sets the critical PCIe settings required by the I210-CS/CL to show up correctly on the PCIe bus with a default device ID.



2. At customer premises (by OEMs), via an Intel provided software tool, which uses a special register set. This tool enables customers to make some customization to the LEDs, device ID, ASPM, etc. and it sets the per-controller settings.

   For security reasons, the I210-CS/CL has a lock-out mechanism after the iNVM is programmed at this stage, to prevent any tampering/retry of the iNVM programming. It is activated by writing a special iNVM word auto-load structure, iNVM word address 0xA, bit 15 set to 1b. The lock-out is active as long as the SECURITY-EN strapping option is enabled.

3. By disabling the SECURITY_EN strapping option, the iNVM lines left blank become writable again like in step 2. This can be useful for fixing iNVM values that were programmed wrongly, or, if boards are resold to a third party who wants to further customize the iNVM. For example, the third party might want a different MAC address or device ID to identify the device with their company's custom software. At the end of this iNVM write cycle, the SECURITY_EN strapping option must be re-enabled.

3.3.3.1 iNVM Programming Flow via Registers

Writing the iNVM via this flow must be done when the system is idle, with no Rx/Tx traffic running and with PCIEMISC.DMA Idle Indication bit set to 1b. The iNVM memory is organized in 32 lines of 64 bits each, for a total of 2 Kb.

1. To be sure the PHY clock used by iNVM programming logic gets stabilized, wait (at least) 15 µs after EEMNGCTL.CFG_DONE bit is read as 1b.
   a. Skip this step on devices that have no attached Flash parts with a valid contents
2. To avoid mistakenly writing the iNVM, write the iNVM_PROTECT.CODE register field with 0xABACADA (ALLOW_WRITE bit is set to 1b).
3. Read the iNVM memory line to be programmed, use iNVM_DATA[2n] and iNVM_DATA[2n+1] register (n=0,...,31), respectively for the lower and higher Dwords of the iNVM line to be programmed.
4. Write the desired value in iNVM_DATA[2n].
5. Wait 320 µs, which is the time required for a complete burning of the 32-bit fuses or poll iNVM_PROTECT.BUSY until it is cleared.
6. Write the desired value in iNVM_DATA[2n+1].
7. Wait 320 µs, which is the time required for a complete burning of the 32-bit fuses.
8. Read the iNVM line programmed via iNVM_DATA[2n] and iNVM_DATA[2n+1] registers read.
   a. If not all the bits were properly written, repeat steps 4 to 8 until all bits are properly written.
9. Optionally, lock the line programmed by setting iNVM_LOCK[n].LOCK register bit to 1b.
   a. Wait 10 µs for the lock to take effect.
   b. Read the iNVM_LOCK[n].LOCK register bit to check it is read as 1b.
   c. If it is not read as 1b, repeat step 9 until it reads as 1b.
10. Program a new line if needed by repeating step 3.
11. When the iNVM programming sequence completes, write to the iNVM_PROTECT register with 0x00000000.

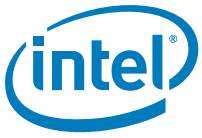
Note: Reading the iNVM can be done directly by read access to the iNVM_DATA[0-63] registers.

Locking a programmed line at step 8 avoids any possibility in the future to invalidate the line by writing the *Type* field with 111b.

In case no Flash part with a valid contents is attached, the new OTP settings will take effect either after a power-up cycle or if mirroring the whole OTP contents into the shadow RAM and initiating a PCIe reset. The later option does not concern items that load only at power-up (refer to Table 3-21).

3.3.4 Hardware Load of iNVM Values into Internal Structures

After every reset, hardware goes over the iNVM, reading and parsing its structures. If a structure is valid and its reset type matches the initiated reset, hardware loads the word or CSR from the iNVM structure into its internal hardware structures.



If an iNVM structure type is read as 111b, hardware skips that Dword, and any following Dword starting with that type field.

If a type field is read as 000b, hardware stops parsing the iNVM and concludes.

In a 2 Kb iNVM, there is room for programming up to 64 words or 32 CSRs. For example, programming the MAC address consumes three auto-load word structures.

Once an iNVM structure is written, there is no way to modify its value other than invalidating its type field (type=111b). iNVM structures can be constructed to rewrite or replace previous iNVM structures, if such a change is required. For instance, assuming PCI configuration and various workarounds require 5 CSR structures and 5 word auto-load ones, then 68 iNVM words remain un-programmed, which leaves enough room for additional word and CSR rewrites, if needed.

3.3.5 Software Load of Default Values into Internal Structures

On every reset event of the I210-CS/CL, software is able to re-load new default values into the internal hardware structures as if the settings were auto-loaded by hardware from the shadow RAM. This ability is referred to as auto-load bus write by software. It is aimed to avoid wasting iNVM lines with settings that can be handled by software.

The following flow is used by software:

1. Write the iNVM word address and data to be loaded in the device via EEARBC register write. Refer to Table 6-1 iNVM words that are used by hardware.
2. Wait until the EEARBC.DONE bit is set by hardware.
3. Load new iNVM words into the hardware structures by repeating steps 1 and 2 as needed.

Note: Software uses the autoload bus write mechanism because writing into registers is not always possible to set internal hardware structures.

3.3.6 I210-CS/CL Init Flow

Once the init flow detects no Flash device is present, a POR or a firmware reset event, the ROM-based firmware code jumps to this flow.

1. If this is the first time this flow is entered after POR, then ROM-firmware parses the iNVM structure to handle PHY register auto-load structures (if there are such in iNVM).
2. ROM-firmware parses the iNVM structure to detect the presence of a word auto-load structure for iNVM word 0x0A:
   a. If the structure is found and HI_DISABLE bit is set to 1b, then exit this flow.
3. ROM-firmware sets the HICR.Memory Base Enable bit to 1b, and HICR.Enable to 1b. This has the effect of enabling the host interface.
4. ROM-firmware sets FWSM.FW_Mode field to 100b (host interface only), the FWSM.FW_Val_Bit to 1b, and issues an ICR.MNG interrupt to the host for notifying it that the device is ready for the proxy code load.
5. ROM-firmware polls the HICR.C bit until it is set to 1b by the host. This is the indication used by the host to notify firmware that the proxy code was loaded.
6. When the software device driver is up, it detects it is a the I211 SKU (device ID read as 0x1539) and it waits until FWSM.FW_Mode is read as 100b (host interface only) and the FWSM.FW_Val_Bit is read as 1b.
7. The software device driver resets the port by setting CTRL.RST and waits for EEC.AUTO_RD to be read as 1b.

8. The software device driver resets the firmware by setting HICR.FWRE to 1b first, and then by setting HICR.FWR to 1b, which has the effect of re-entering the ROM-firmware into step 1.
9. Each time the system exits from a sleep state, or once the software device driver gets the interrupt issued by firmware at step 2, the software device driver checks whether the FWSM.FW_Mode is read as 100b (host interface only) and the FWSM.FW_Val_Bit is read as 1b. This is the method used by firmware to request re-load of the proxy code.
10. If a proxy code has to be loaded, then the software device driver sets its current internal RAM base address to 0x10000. Otherwise, the software device driver exits the flow.
11. The software device driver copies its current internal RAM base address into the HIBBA register.
12. The software device driver writes consecutive locations from address 0x8800 up to 0x8BFF with the next 1 KB of the proxy code, in Dwords (32-bit) chunks ordered in little endian.
13. The software device driver increments its current internal RAM base address by 1 KB.
14. The software device driver repeats steps 10 to 12 until the entire proxy code is written (or until the 50 KB limit is reached).
15. The software device driver sets the HICR.C bit to notify the ROM-firmware that the proxy code load completed.
16. ROM-firmware starts the proxy code execution from internal RAM address 0x10000.
17. Once RAM-firmware completes it's init sequence and is ready to receive commands from host, it sets FWSM.FW_Mode to 001b (the I211 mode) and the FW_Val_Bit to 1b, and it clears the HICR.C bit to notify the host that the host interface is ready to receive commands.

Note: Once loaded, the firmware runs the proxy code even when the system is in a sleep state. It is the software device driver's responsibility to reset the firmware prior to entering Sx.

However, if the system powers up in S3 state or if a firmware reset event occurs while the system was in S3, no proxy offload is performed until the system resumes S0 and the proxy code is re-loaded into the device.

After PHY reset events, ROM-firmware (as well as RAM-firmware) is responsible to parse the PHY register auto-load structures of the iNVM (refer to Section 3.3.2.3) and to perform the required MDIO accesses accordingly.

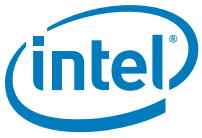
3.4 Configurable I/O Pins

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

The I210-CS/CL 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 Flash 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 Flash 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 3.4.3), IEEE 1588 support (refer to Section 6.8) or an I^2C interface bus (refer to Section 3.4.2).

3.4.2 I^2C Over SDP

The I^2C usage of SDP pins must be enabled by setting the I2C_ON_SDP_EN bit to 1b in Flash word 0x20. This relates to the SDP 0 and SDP 2 pins, which operate as I2C_CLK and I2C_DATA, respectively.

The I^2C interface operates via the I2CCMD and I2CPARAMS register set (refer to Section 7.17.8). Since this register set can be used by either software or firmware in alternation, its ownership must be acquired/released via the semaphore ownership taking/release flows described in Section 4.6.

Note: I^2C over SDP pins mode is mutually exclusive with running I^2C over the SFPx_I2C pins.

3.4.3 LEDs

The I210-CS/CL provides three LEDs on the port that can be used to indicate different statuses of the traffic. The default setup of the LEDs is done via Flash word offsets 0x1C and 0x1F. This setup is reflected in the LEDCTL register. Each software device driver can change its setup individually. For each of the LEDs, the following parameters can be defined:

- Mode: Defines which information is reflected by this LED. The encoding is described in the LEDCTL register.
- Polarity: Defines the polarity of the LED.
- Blink mode: Determines whether or not the LED should blink or be stable.

In addition, the blink rate of all LEDs can be defined. The possible rates are 200 ms or 83 ms for each phase. There is one rate for all the LEDs.

3.5 Voltage Regulator

To reduce Bill of Material (BOM) cost, the I210-CS/CL supports generating the 1.5V and 0.9V power supplies from the 3.3V supply using an on-chip Switching Capacitor Voltage Regulator (SVR) control circuit, which requires only an external capacitor component.

Refer to Section 9.6.6 for more details.

3.6 Network Interfaces

3.6.1 Overview

The I210-CS/CL MAC provides 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) implementations. The I210-CS/CL performs all of the functions required for transmission, reception, and collision handling called out in the standards.

The I210-CS/CL supports the following potential configurations:

- External SerDes device such as an optical SerDes (SFP or on board) or backplane (1000BASE-BX or 1000BASE-KX) connections.
- External SGMII device. This mode is used for connections to external 10/100/1000 BASE-T PHYs that support the SGMII MAC/PHY interface.

Selecting between the various configurations is programmable via the MAC's the Extended Device Control register (*CTRL_EXT.LINK_MODE* bits) and default is set via Flash settings. Table 3-16 lists the encoding on the *LINK_MODE* field for each of the modes.

Table 3-16. Link Mode Encoding

Link Mode	I210-CS/CL Mode
00b	Reserved
01b	1000BASE-KX
10b	SGMII
11b	SerDes/1000BASE-BX

The GMII/MII interface, used to communicate between the MAC and the external PHY or the SGMII PCS, supports 10/100/1000 Mb/s operation, with both half- and full-duplex operation at 10/100 Mb/s, and only full-duplex operation at 1000 Mb/s. Board design link path should include AC coupling capacitors.

The SerDes function can be used to implement a fiber-optics-based solution or backplane connection without requiring an external TBI mode transceiver/SerDes.

The SerDes interface can be used to connect to SFP modules. As such, this SerDes interface has the following limitations:

- No Tx clock
- AC coupling only

3.6.2 MAC Functionality

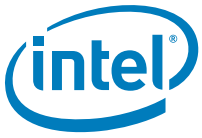
3.6.2.1 MDIO/MDC PHY Management Interface

The I210-CS/CL implements an IEEE 802.3 MII Management Interface, also known as the Management Data Input/Output (MDIO) or MDIO interface, between the MAC and a PHY. This interface provides the MAC and software the ability to monitor and control the state of the PHY. The MDIO interface defines a physical connection, a special protocol that runs across the connection, and an internal set of addressable registers. The interface consists of a 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 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 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 SerDes or an external SGMII PHY, by accessing the I210-CS/CL's MDIC register (refer to Section 7.2.4). MDIO configuration setup (external PHY, PHY address and shared MDIO) is defined in the MDICNFG register (refer to Section 7.2.5). By selecting Page 26 via PHYREG 22 register, internal SerDes registers can be accessed.

When working in SGMII/SerDes mode, the external PHY (if it exists) can be accessed either through MDC/MDIO as previously described, or via a two wire I^2C interface bus using the I2CCMD register (refer to Section 7.17.8). The two wire I^2C interface bus or the MDC/MDIO bus are connected via the same



pins, and thus are mutually exclusive. In order to be able to control an external device, either by I^2C or MDC/MDIO, the 2-wires *SFP Enable* bit in Initialization Control 3 Flash word, that's loaded into the *CTRL_EXT.I2C Enabled* register bit, should be set.

The external port PHY address is written in the *MDICNFG.PHYADD* register field, which is loaded from the Initialization Control 4 Flash word following reset.

Note: When the dedicated SFPx_I2C pins are not used for I^2C, an alternative I^2C interface bus can be run over SDP 0 and SDP2 pins, using the *I2CCMD* register set (refer to Section 3.4.2).

3.6.2.1.1 Detecting an External I^2C or MDIO Connection

When the *CTRL_EXT.I2C Enabled* bit is set to 1b, software can recognize type of external PHY control bus (MDIO or I^2C) connection according to the values loaded from the Flash to the *MDICNFG.Destination* bit and the *CTRL_EXT.LINK_MODE* field in the following manner:

- External I^2C operating mode - *MDICNFG.Destination* equals 0b and *CTRL_EXT.LINK_MODE* is not equal to 0b.
- External MDIO Operating mode - *MDICNFG.Destination* equals 1b and *CTRL_EXT.LINK_MODE* is not equal to 0b.

3.6.2.1.2 MDIC and MDICNFG Register Usage

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

1. If default *MDICNFG* register values need to be updated, the processor performs a PCIe write access to the *MDICNFG* register to define the:
   — PHYADD = Address of external PHY.
   — Destination = External PHY.
2. 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).
3. The MAC applies the following sequence on the MDIO signal to the PHY:

   <PREAMBLE><01><10><PHYADD><REGADD><Z> where Z stands for the MAC tri-stating the MDIO signal.
4. The PHY returns the following sequence on the MDIO signal<0><DATA><IDLE>.
5. The MAC discards the leading bit and places the following 16 data bits in the MII register.
6. The I210-CS/CL asserts an interrupt indicating MDIO Done if the *Interrupt Enable* bit was set.
7. The I210-CS/CL sets the *Ready* bit in the MDIC register indicating the read completed.
8. 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:

1. If default MDICNFG register values loaded need to be updated, the processor performs a PCIe write cycle to the MDICNFG register to define the:
   — PHYADD = Address of external PHY.
   — Destination = External PHY.

2. 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.
3. The MAC applies the following sequence on the MDIO signal to the PHY:

   <PREAMBLE><01><01><PHYADD><REGADD><10><DATA><IDLE>
4. The I210-CS/CL asserts an interrupt indicating MDIO Done if the *Interrupt Enable* bit was set.
5. The I210-CS/CL sets the *Ready* bit in the MDIC register to indicate that the write operation completed.
6. 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 should not be attempted for 300 μs.

3.6.2.2 Duplex Operation with Copper PHY

The I210-CS/CL supports half-duplex and full-duplex 10/100 Mb/s MII mode or SGMII interface. However, only full-duplex mode is supported when SerDes/1000BASE-BX or 1000BASE-KX modes are used or in any 1000 Mb/s connection.

Configuring the I210-CS/CL duplex operation can either be forced or determined via the auto-negotiation process.

3.6.2.2.1 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 the I210-CS/CL and the specific capabilities of the link partner used in the application. During full-duplex operation, the I210-CS/CL 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 PCS. Reception is signaled by the PCS by the asserting the RX_DV signal, which indicates valid receive data on the RX_DATA lines to the MAC.

3.6.2.2.2 Half Duplex

In half-duplex operation, the MAC attempts to avoid contention with other traffic on the link by monitoring the CRS signal provided by the PCS and deferring to passing traffic. When the CRS signal is de-asserted or after a sufficient Inter-Packet Gap (IPG) has elapsed after a transmission, frame transmission begins. The MAC signals the PCS with TX_EN at the start of transmission.

In the case of a collision, the SGMII detects the collision and asserts the COL signal to the MAC. Frame transmission stops within four link clock times and then the I210-CS/CL sends a JAM sequence onto the link. After the end of a collided transmission, the I210-CS/CL backs off and attempts to re-transmit per the standard CSMA/CD method.

Note: The re-transmissions are done from the data stored internally in the I210-CS/CL MAC transmit packet buffer (no re-access to the data in host memory is performed).

The MAC behavior is different if a regular collision or a late collision is detected. If a regular collision is detected, the MAC always tries to re-transmit until the number of excessive collisions is reached. In case of late collision, the MAC retransmission is configurable. In addition, statistics are gathered on late collisions.

In the case of a successful transmission, the I210-CS/CL is ready to transmit any other frame(s) queued in the MAC's transmit FIFO, after the minimum inter-frame spacing (IFS) of the link has elapsed.

During transmit, the PCS is expected to signal a carrier-sense (assert the CRS signal) back to the MAC before one slot time has elapsed. The transmission completes successfully even if the PCS fails to indicate CRS within the slot time window. If this situation occurs, the PCS can either be configured incorrectly or be in a link down situation. Such an event is counted in the transmit without CRS statistic register (refer to Section 7.18.12).

3.6.3 SerDes/1000BASE-BX, SGMII and 1000BASE-KX Support

The I210-CS/CL can be configured to follow either SGMII, SerDes/1000BASE-BX or 1000BASE-KX standards. When in SGMII mode, the I210-CS/CL can be configured to operate in 1 Gb/s, 100 Mb/s or 10 Mb/s speeds. When in the 10/100 Mb/s speed, the I210-CS/CL can be configured to half-duplex mode of operation. When configured for SerDes/1000BASE-BX or 1000BASE-KX operation, the port supports only 1 Gb/s, full-duplex operation. Since the serial interfaces are defined as differential signals, internally the hardware has analog and digital blocks. Following is the initialization/ configuration sequence for the analog and digital blocks.

3.6.3.1 SerDes/1000BASE-BX, SGMII and 1000BASE-KX Analog Block

The analog block might require some changes to its configuration registers in order to work properly. There is no special requirement for designers to do these changes as the hardware internally updates the configuration using a default sequence or a sequence loaded from the Flash.

3.6.3.2 SerDes/1000BASE-BX, SGMII and 1000BASE-KX PCS Block

The link setup for SerDes/1000BASE-BX, 1000BASE-KX and SGMII are described in sections 3.6.4.1, 3.6.4.2 and 3.6.4.3 respectively.

3.6.3.3 GbE Physical Coding Sub-Layer (PCS)

The I210-CS/CL integrates the 802.3z PCS function on-chip. The on-chip PCS circuitry is used when the link interface is configured for SerDes/1000BASE-BX, 1000BASE-KX or SGMII operation.

The packet encapsulation is based on the Fiber Channel (FC0/FC1) physical layer and uses the same coding scheme to maintain transition density and DC balance. The physical layer device is the SerDes and is used for 1000BASE-SX, -L-, or -CX configurations.

3.6.3.3.1 8B10B 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.

3.6.3.3.2 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 3-17. 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. The transitions in the idle stream enable the SerDes to maintain clock and symbol synchronization between link partners.
/I1/	IDLE 1	2	See /I/. Differs from /I2/ in 2nd code group for maintaining proper signaling disparity[1].

Table 3-17. Brief Description of Defined Ordered Sets (Continued)

Code	Ordered_Set	# of Code Groups	Usage
/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.

3.6.4 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 or SerDes). In SerDes/1000BASE-BX mode, the I210-CS/CL provides the complete PCS and Auto-negotiation functionality as defined in IEEE802.3 clause 36 and clause 37. In SGMII mode, the I210-CS/CL supports the SGMII link auto-negotiation process, whereas the link auto-negotiation, as defined in IEEE802.3 clause 28 and clause 40, is done by the external PHY. In 1000BASE-KX mode, the I210-CS/CL supports only parallel detect of 1000BASE-KX signaling and does not support the full Auto-Negotiation for Backplane Ethernet protocol as defined in IEEE802.3ap clause 73.

Configuring the link can be accomplished by several methods ranging from software forcing link settings, software-controlled negotiation, MAC-controlled auto-negotiation, to auto-negotiation initiated by a PHY. The following sections describe processes of bringing the link up including configuration of the I210-CS/CL and the transceiver, as well as the various methods of determining duplex and speed configuration.

The process of determining link configuration differs slightly based on the specific link mode (SerDes/1000BASE-BX, SGMII or 1000BASE-KX) being used.

When operating in a SerDes/1000BASE-BX mode, the PCS layer performs auto-negotiation per clause 37 of the 802.3z standard. The transceiver used in this mode does not participate in the auto-negotiation process as all aspects of auto-negotiation are controlled by the I210-CS/CL.

When operating in SGMII mode, the PCS layer performs SGMII auto-negotiation per the SGMII specification. The external PHY is responsible for the Ethernet auto-negotiation process.

When operating in 1000BASE-KX mode the I210-CS/CL performs parallel detect of 1000BASE-KX operation but does not implement the full auto-negotiation for backplane Ethernet sequence as defined in IEEE802.3ap clause 73.

3.6.4.1 SerDes/1000BASE-BX Link Configuration

When using SerDes/1000BASE-BX link mode, link mode configuration can be performed using the PCS function in the I210-CS/CL. The hardware supports both hardware and software auto-negotiation methods for determining the link configuration, as well as allowing for a manual configuration to force the link. Hardware auto-negotiation is the preferred method.

3.6.4.1.1 Signal Detect Indication

When the *CONNSW.ENRGSRC* bit is set to 1b, the SRDS_SIG_DET pins can be connected to a Signal Detect or loss-of-signal (LOS) output of the optical module that indicates when no laser light is being received when the I210-CS/CL is used in a 1000BASE-SX or -LX implementation (SerDes operation). No standard polarity for the signal detect or loss-of-signal driven from different manufacturer optical modules exists. The *CTRL.ILOS* bit provides the capability to invert the signal from different external optical module vendors, and should be set when the external optical module provides a negative-true loss-of-signal.

Note: In SGMII, 1000BASE-BX and 1000BASE-KX connections, energy detect source is always internal and value of *CONNSW.ENRGSRC* bit should be 0b. The *CTRL.ILOS* bit also inverts the internal link-up input that provides link status indication and thus should be set to 0b for proper operation.

3.6.4.1.2 MAC Link Speed

SerDes/1000BASE-BX operation is only defined for 1000 Mb/s operation. Other link speeds are not supported. When configured for the SerDes interface, the MAC speed-determination function is disabled and the Device Status register bits (*STATUS.SPEED*) indicate a value of 10b for 1000 Mb/s.

3.6.4.1.3 SerDes/1000BASE-BX Mode Auto-Negotiation

In SerDes/1000BASE-BX mode, after power up or the I210-CS/CL reset via PE_RST_N, the I210-CS/CL initiates IEEE802.3 clause 37 auto-negotiation based on the default settings in the device control and transmit configuration or PCS Link Control Word registers, as well as settings read from the Flash. If enabled in the Flash, the I210-CS/CL immediately performs auto-negotiation.

TBI mode auto-negotiation, as defined in clause 37 of the IEEE 802.3z standard, provides a protocol for two devices to advertise and negotiate a common operational mode across a GbE link. The I210-CS/CL fully supports the IEEE 802.3z auto-negotiation function when using the on-chip PCS and internal SerDes.

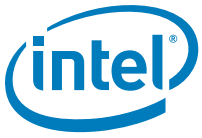
TBI mode auto-negotiation is used to determine the following information:

- Duplex resolution (even though the I210-CS/CL MAC only supports full-duplex in SerDes/ 1000BASE-BX mode).
- Flow control configuration.

Note: Since speed for SerDes/1000BASE-BX modes is fixed at 1000 Mb/s, speed settings in the Device Control register are unaffected by the auto-negotiation process.

Note: Auto-negotiation can be initiated at power up or by asserting PE_RST_N and enabling specific bits in the Flash.

The auto-negotiation process is accomplished by the exchange of /C/ ordered sets that contain the capabilities defined in the *PCS_ANADV* register in the 3rd and 4th symbols of the ordered sets. Next page are supported using the *PCS_NPTX_AN* register.



Bits *FD* and *LU* in the Device Status (STATUS) register, and bits in the *PCS_LSTS* register provide status information regarding the negotiated link.

Auto-negotiation can be initiated by the following:

- *PCS_LCTL.AN_ENABLE* transition from 0b to 1b
- Receipt of /C/ ordered set during normal operation
- Receipt of a different value of the /C/ ordered set during the negotiation process
- Transition from loss of synchronization to synchronized state (if *AN_ENABLE* is set).
- *PCS_LCTL.AN_RESTART* transition from 0b to 1b

Resolution of the negotiated link determines device operation with respect to flow control capability and duplex settings. These negotiated capabilities override advertised and software-controlled device configuration.

Software must configure the *PCS_ANADV* fields to the desired advertised base page. The bits in the Device Control register are not mapped to the *txConfigWord* field in hardware until after auto-negotiation completes. Table 3-18 lists the mapping of the *PCS_ANADV* fields to the Config_reg Base Page encoding per clause 37 of the standard.

Table 3-18. 802.3z Advertised Base Page Mapping

15	14	13:12	11:9	8:7	6	5	4:0
Nextp	Ack	RFLT	rsv	ASM	Hd	Fd	rsv

The partner advertisement can be seen in the PCS_LPAB and PCS_LPABNP registers.

3.6.4.1.4 Forcing Link-up in SerDes/1000BASE-BX Mode

Forcing link can be accomplished by software by writing a 1b to *CTRL.SLU*, which forces the MAC PCS logic into a link-up state (enables listening to incoming characters when SRDS_[n]_SIG_DET is asserted by the external optical module.

Note: The *PCS_LCTL.AN_ENABLE* bit must be set to a logic zero to enable forcing link.

When link is forced via the *CTRL.SLU* bit, the link does not come up unless the SRDS_[n]_SIG_DET signal is asserted or an internal energy indication is received from the SerDes receiver, implying that there is a valid signal being received by the optical module or SerDes circuitry.

The source of the signal detect is defined by the *ENRGSRC* bit in the CONNSW register.

3.6.4.1.5 Hardware Detection of Non-Auto-Negotiation Partner

Hardware can detect a SerDes link partner that sends idle code groups continuously, but does not initiate or answer an auto-negotiation process. In this case, hardware initiates an auto-negotiation process, and if it fails after some timeout, a link up is assumed. To enable this functionality the *PCS_LCTL.AN_TIMEOUT_EN* bit should be set. This mode can be used instead of the force link mode as a way to support a partner that do not support auto-negotiation.

3.6.4.2 1000BASE-KX Link Configuration

When using 1000BASE-KX link mode, link mode configuration is forced manually by software since the I210-CS/CL does not support IEEE802.3 clause 73 backplane auto-negotiation.

3.6.4.2.1 MAC Link Speed

1000BASE-KX operation is only defined for 1000 Mb/s operation. Other link speeds are not supported. When configured for the 1000BASE-KX interface, the MAC speed-determination function is disabled and the Device Status register bits (*STATUS.SPEED*) indicate a value of 10b for 1000 Mb/s.

3.6.4.2.2 1000BASE-KX Auto-Negotiation

The I210-CS/CL only supports parallel detection of the 1000BASE-KX link and does not support the full IEEE802.3ap clause 73 backplane auto-negotiation protocol.

3.6.4.2.3 Forcing Link-up in 1000BASE-KX Mode

In 1000BASE-KX mode (*EXT_CTRL.LINK_MODE* = 01b) the I210-CS/CL should always operates in force link mode (*CTRL.SLU* bit is set to 1b). The MAC PCS logic is placed in a link-up state once energy indication is received, implying that a valid signal is being received by the 1000BASE-KX circuitry. When in the link-up state PCS logic can lock on incoming characters.

Note: In 1000BASE-KX mode energy detect source is internal and value of *CONNSW.ENRGSRC* bit should be 0b. Clause 37 auto-negotiation should be disabled and the value of the *PCS_LCTL.AN_ENABLE* bit and *PCS_LCTL.AN TIMEOUT EN* bit should be 0b.

3.6.4.2.4 1000BASE-KX Hardware Detection of Link Partner

In 1000BASE-KX mode, hardware detects a 1000BASE-KX link partner that sends idle or none idle code groups continuously. In 1000BASE-KX operation force link-up mode is used.

3.6.4.3 SGMII Link Configuration

When working in SGMII mode, the actual link setting is done by the external PHY and is dependent on the settings of this PHY. The SGMII auto-negotiation process described in the sections that follow is only used to establish the MAC/PHY connection.

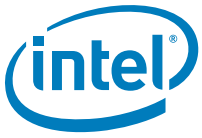
3.6.4.3.1 SGMII Auto-Negotiation

This auto-negotiation process is not dependent on the SRDS_[n]_SIG_DET signal and the *CONNSW.ENRGSRC* bit should be 0b, as this signal indicates optical module signal detection and is not relevant in SGMII mode.

The outcome of this auto-negotiation process includes the following information:

- Link status
- Speed
- Duplex

This information is used by hardware to configure the MAC, when operating in SGMII mode.



Bits *FD* and *LU* of the Device Status (STATUS) register and bits in the PCS_LSTS register provide status information regarding the negotiated link.

Auto-negotiation can be initiated by the following:

- *PCS_LCTL.AN_ENABLE* transition from 0b to 1b.
- Receipt of /C/ ordered set during normal operation.
- Receipt of different value of the /C/ ordered set during the negotiation process.
- Transition from loss of synchronization to a synchronized state (if *AN_ENABLE* is set).
- *PCS_LCTL.AN_RESTART* transition from 0b to 1b.

Auto-negotiation determines the I210-CS/CL operation with respect to speed and duplex settings. These negotiated capabilities override advertised and software controlled device configuration.

When working in SGMII mode, there is no need to set the PCAS_ANADV register, as the MAC advertisement word is fixed. In SGMII mode the *PCS_LCTL.AN TIMEOUT EN* bit should be 0b, since auto-negotiation outcome is required for correct operation. The result of the SGMII level auto-negotiation can be read from the PCS_LPAB register.

3.6.4.3.2 Forcing Link in SGMII Mode

In SGMII, forcing of the link cannot be done at the PCS level, only in the external PHY. The forced speed and duplex settings are reflected by the SGMII auto-negotiation process; the MAC settings are automatically done according to this functionality.

3.6.4.3.3 MAC Speed Resolution

The MAC speed and duplex settings are always set according to the SGMII auto-negotiation process.

3.6.4.4 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. When the MAC is configured for optical SerDes mode, the input reflects loss-of-signal connection from the optics. In backplane mode, where there is no LOS external indication, an internal indication from the SerDes receiver can be used. In SFP systems the LOS indication from the SFP can be used. Assuming that the MAC has been configured with *CTRL.SLU*=1b, the MAC status bit *STATUS.LU*, when read, generally reflects whether SerDes has link.

When the link indication from the loss-of-signal asserted from the SerDes, 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.

3.6.5 Ethernet Flow Control (FC)

The I210-CS/CL 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.
- DMA Coalescing Receive Threshold High (FCRTC) - A 13-bit high watermark indicating receive buffer fullness when in DMA coalescing and Tx buffer is empty. The value in this register can be higher than value placed in the FCRTH0 register since the watermark needs to be set to allow for only receiving a maximum sized Rx packet before XOFF flow control takes effect and reception is stopped (refer to Table 3-22 for information on flow control threshold calculation).
- 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

3.6.5.1 MAC Control Frames and Receiving Flow Control Packets

3.6.5.1.1 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 the I210-CS/CL (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.

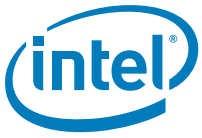


Table 3-19 lists the structure of a 802.3X FC packet.

Table 3-19. 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

3.6.5.1.2 Operation and Rules

The I210-CS/CL 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, the I210-CS/CL 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.

3.6.5.1.3 Timing Considerations

When operating at 1 Gb/s line speed, the I210-CS/CL 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, the I210-CS/CL 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.

3.6.5.2 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 3-20 lists the behavior of the *DPF* bit.

Table 3-20. 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 6.1.1.1).[1]
1	0	Yes.
1	1	No.

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

Table 3-21 defines the behavior of the *PMCF* bit.

Table 3-21. 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 6.1.1.1).
1	1	Yes.
1	0	No.

3.6.5.3 Transmission of PAUSE Frames

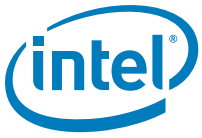
The I210-CS/CL generates PAUSE packets to ensure there is enough space in its receive packet buffers to avoid packet drop. The I210-CS/CL monitors the fullness of its receive packet buffers and compares it with the contents of a programmable threshold. When the threshold is reached, the I210-CS/CL sends a PAUSE frame. The I210-CS/CL 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.

The transmission of flow control frames should only be enabled in full-duplex mode per the IEEE 802.3 standard. Software should ensure that the transmission of flow control packets is disabled when the I210-CS/CL is operating in half-duplex mode.

3.6.5.3.1 Operation and Rules

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



The I210-CS/CL sends a PAUSE frame when Rx packet buffer is full above the high threshold defined in the Flow Control Receive Threshold High (*FCRT0.RTH*) register field. When the threshold is reached, the I210-CS/CL 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 *FCRT0.RTH* value are listed in Table 3-22.

Table 3-22. 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	Max packet size.	Max packet size * 1.25.
XOFF Reception to Link Partner response	Max packet size.	Max packet size.
Round trip latency	The latencies on the wire and the LAN devices at both sides of the wire.	320-byte (for 1000Base-T operation).

Note: When DMA Coalescing is enabled (*DMACR.DMAC_EN* = 1b),the value placed in the *FCRTC.RTH_Coal* field should be equal or lower than:

FCRTC.RTH_Coal = *FCRTH0.RTH* + Max packet size * 1.25

The *FCRTC.RTH_Coal* is used as the high watermark to generate XOFF flow control packets when the internal Tx buffer is empty and the I210-CS/CL is executing DMA coalescing. In this case, no delay to transmission of flow control packet exists so its possible to increase level of watermark before issuing a XOFF flow control frame.

After transmitting a PAUSE frame, the I210-CS/CL 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, the I210-CS/CL sends a XON message (a PAUSE frame with a timer value of zero). Software enables this capability with the *XONE* field of FCRTL.

The I210-CS/CL 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.

3.6.5.3.2 Software Initiated PAUSE Frame Transmission

The I210-CS/CL 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.

XOFF transmission is not supported in 802.3x for half-duplex links. Software should not initiate an XOFF or XON transmission if the I210-CS/CL is configured for half-duplex operation.

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).

3.6.5.4 IPG Control and Pacing

The I210-CS/CL supports the following modes of controlling IPG duration:

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

3.6.5.4.1 Fixed IPG Extension

The I210-CS/CL 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 the I210-CS/CL without violating the minimum IPG requirements. For example, a security device connected in series to the I210-CS/CL might add security headers to transmit packets before the packets are transmitted on the network.

3.6.6 Loopback Support

3.6.6.1 General

The I210-CS/CL supports the following types of internal loopback in the LAN interface:

- MAC Loopback (Point 1)
- SerDes, SGMII or 1000BASE-KX Loopback (Point 3)

By setting the device to loopback mode, packets that are transmitted towards the line are looped back to the host. The I210-CS/CL is fully functional in these modes, just not transmitting data over the lines. Figure 3-5 shows the points of loopback.

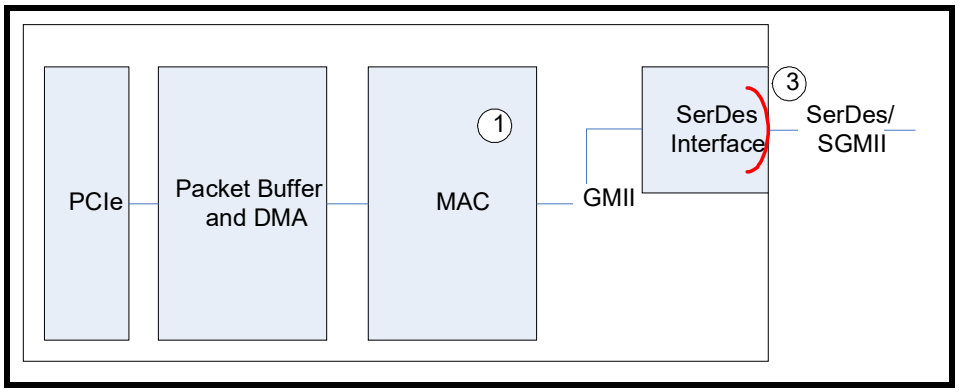


Figure 3-5. I210-CS/CL Loopback Mode

3.6.6.2 MAC Loopback

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

3.6.6.2.1 Setting the I210-CS/CL to MAC loopback Mode

The following procedure should be used to put the I210-CS/CL 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 10b (1 GbE).

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

3.6.6.3 SerDes, SGMII and 1000BASE-KX Loopback

In SerDes, SGMII or 1000BASE-KX loopback, the PHY block is not functional and data is looped back at the end of the relevant functionality. This means all designs that are functional in SerDes/SGMII or 1000BASE-KX mode, are involved in the loopback.

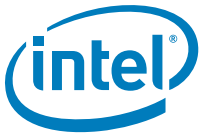
Note: SerDes loopback is functional only if the SerDes link is up.

3.6.6.3.1 Setting the I210-CS/CL to SerDes/1000BASE-BX, SGMII or 1000BASE-KX Loopback Mode

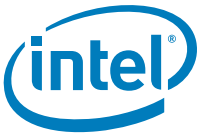
The following procedure should be used to place the I210-CS/CL in SerDes loopback mode:

- Set Link mode to either SerDes, SGMII or 1000BASE-KX by:
  - 1000BASE-KX: *CTRL_EXT.LINK_MODE* = 01b
  - SGMII: *CTRL_EXT.LINK_MODE* = 10b
  - SerDes/1000BASE-BX: *CTRL_EXT.LINK_MODE* = 11b
- Configure SerDes to loopback: *RCTL.LBM* = 11b

- Move to Force mode by setting the following bits:
  - *CTRL.FD* (CSR 0x0 bit 0) = 1b
  - *CTRL.SLU* (CSR 0x0 bit 6) = 1b
  - *CTRL.RFCE* (CSR 0x0 bit 27) = 0b
  - *CTRL.TFCE* (CSR 0x0 bit 28) = 0b
  - *PCS_LCTL.FORCE_LINK* (CSR 0X4208 bit 5) = 1b
  - *PCS_LCTL.FSD* (CSR 0x4208 bit 4) = 1b
  - *PCS_LCTL.FDV* (CSR 0x4208 bit 3) = 1b
  - *PCS_LCTL.FLV* (CSR 0x4208 bit 0) = 1b
  - *PCS_LCTL.AN_ENABLE* (CSR 0x4208 bit 16) = 0b



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



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 the I210-CS/CL is ready to accept host commands.

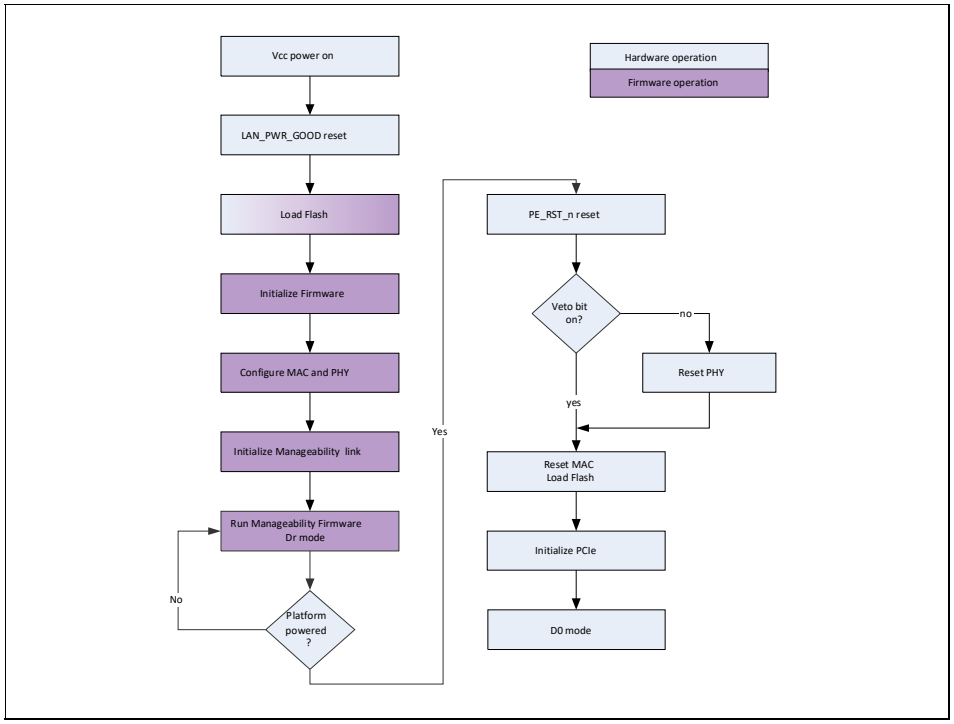


Figure 4-1. Power-Up - General Flow

4.1.2 Power-Up Timing Diagram

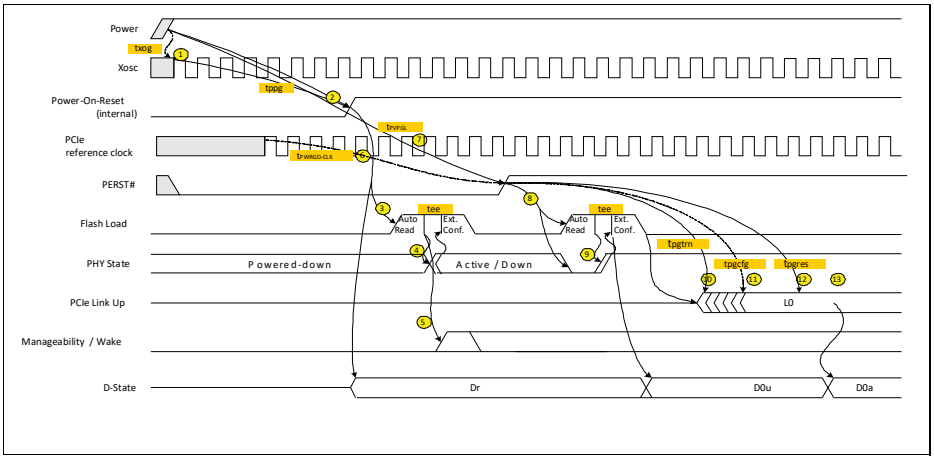


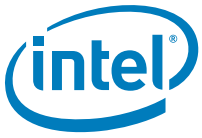
Figure 4-2. Power-Up Timing Diagram

Table 4-1. Notes to Power-Up Timing Diagram

Note	
1	Xosc is stable t_{xog} after the Power is stable
2	Internal Reset is released after all power supplies are good and t_{ppg} after Xosc is stable.
3	A Flash read starts on the rising edge of the internal Reset or LAN_PWR_GOOD.
4	After reading the Flash, the PHY might exit power down mode.
5	APM Wakeup might be enabled based on Flash contents.
6	The PCIe reference clock is valid $t_{PE\_RST-CLK}$ before the de-assertion of PE_RST# (according to PCIe specification).
7	PE_RST# is de-asserted t_{PVPGL} after power is stable (according to PCIe specification).
8	De-assertion of PE_RST# causes the Flash to be re-read, asserts PHY power-down (except if the *Veto* bit also known as Keep_PHY_Link_Up bit is set), and disables Wake Up.
9	After reading the Flash the PHY exits power-down mode.
10	Link training starts after t_{pgtrn} from PE_RST# de-assertion.
11	A first PCIe configuration access might arrive after t_{pgcfg} from PE_RST# de-assertion.
12	A first PCI configuration response can be sent after tpgres from PE_RST# de-assertion
13	Writing a 1b to the *Memory Access Enable* bit in the PCI *Command Register* transitions the device from D0u to D0 state.

4.2 Reset Operation

The I210-CS/CL has a number of reset sources described in the sections that follow. After a reset, the software device driver should verify that the *EEMNGCTL.CFG_DONE* bit (refer to Section 7.4.15) is set to 1b and no errors were reported in the *FWSM.Ext_Err_Ind* (refer to Section 7.7.2) field.



4.2.1 Reset Sources

The I210-CS/CL reset sources are described in the sections that follow.

4.2.1.1 LAN_PWR_GOOD

The I210-CS/CL has an internal mechanism for sensing the power pins. Once power is up and stable, the I210-CS/CL creates an internal 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. LAN_PWR_GOOD is interpreted to be an indication that device power supplies are all stable. Note that LAN_PWR_GOOD changes state during system power-up.

4.2.1.2 PE_RST_N

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

The I210-CS/CL 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 PCI reset in previous (PCI) GbE LAN controllers.

4.2.1.4 D3hot to D0 Transition

This is also known as ACPI reset. The I210-CS/CL 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 the I210-CS/CL resets internal CSRs (similar to *CTRL.RST* assertion) but doesn't reset registers in the PCIe configuration space. If the *PMCSR.No_Soft_Reset* bit is cleared, the I210-CS/CL 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 the I210-CS/CL to D3.

4.2.1.5 FLR

A FLR function reset is issued by setting bit *15* in the *Device Control* configuration register (refer to Section 8.4.5.4), 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 Flash content is partially reloaded after a FLR reset. The words read from Flash at FLR are the same as read following a full software reset.

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 8.4.5.5), 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.

4.3 Software Reset

4.3.1 Software Reset (RST)

Software can reset the I210-CS/CL by setting the Software Reset (*CTRL.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 operate the master disable algorithm as defined in Section 5.2.3.3.

The *CTRL.RST* bit is provided primarily to recover from an indeterminate or suspected port hung hardware state. Most registers (receive, transmit, interrupt, statistics, etc.) and state machines in the port are set to their power-on reset values, approximating the state following a power-on or PCIe reset (refer to Table 4-3 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.

Note: To ensure that a software reset fully completed and that the I210-CS/CL responds correctly to subsequent accesses after setting the *CTRL.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.PF_RST_DONE* bit is set to 1b.

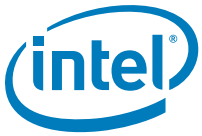
When asserting the *CTRL.RST* software reset bit, only some Flash bits related to the specific function are re-read. Bits re-read from Flash are reset to default values.

4.3.1.1 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 related to the port. The master disable resets only the DMA activities related to this function without affecting activity of other functions or LAN ports. Configuration accesses and target accesses to the function are still enabled and the Management Controller (MC) can still transmit and receive packets on the port.

A Master Disable resets all the queues and DMA related interrupts. It also disables the transmit and receive flows. All pending read requests are dropped and PCIe read completions to this function might be completed as unexpected completions and silently discarded (following update of flow control credits) without logging or signaling it as an error.

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.



4.3.1.2 Flash Reset

Writing a 1b to the Flash *Reset* bit of the Extended Device Control Register (*CTRL_EXT.EE_RST*) causes the I210-CS/CL to re-read the per-function configuration from the Flash, setting the appropriate bits in the registers loaded by the Flash.

4.3.2 Registers and Logic Reset Affects

The resets affect the following registers and logic:

Table 4-2. I210 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)	X	X	X		
PCIe Link Data Path	X	X	X		
Read Flash					14.
Read Flash (Complete Load)	X	X	X		
PCI Configuration Registers - Non Sticky	X	X	X		2.
PCI Configuration Registers - Sticky	X	X	X		3.
PCIe Local Registers	X	X	X		4.
Data Path	X	X	X		
On-die Memories	X	X	X		11.
MAC, PCS, Auto Negotiation and Other Port Related Logic	X	X	X		
DMA	X	X	X		
Functions Queue Enable	X	X	X		
Function interrupt and Statistics Registers	X	X	X		
Wake Up (PM) Context	X				6.
Wake Up Control Register	X				7.
Wake Up Status Registers	X				8.
MMS Unit	X			X	
Wake-Up Management Registers	X	X	X		2.,9.
Memory Configuration Registers	X	X	X		2.
Flash Requests	X				12.
PHY/SerDes PHY	X	X	X		2.
Strapping Pins	X	X	X		

Table 4-3. I210-CS/CL Reset Affects - Other Resets

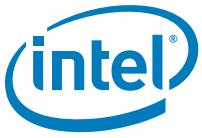
Reset Activation	D3hot –> D0	FLR	Port SW Reset (CTRL.RST)	Force TCO	EE Reset	Notes
Port Configuration Autoload from Flash	X	X	X	X	X	
PCI Configuration Registers Read Only						2.
PCI Configuration Registers MSI-X	X	X				5.
PCI Configuration Registers Read/Write						

Table 4-3. I210-CS/CL Reset Affects - Other Resets (Continued)

Reset Activation	D3hot -> D0	FLR	Port SW Reset (CTRL.RST)	Force TCO	EE Reset	Notes
PCIe Local Registers						4
Data Path	X	X	X	X		
On-die Memories	X	X	X	X		11
MAC, PCS, Auto Negotiation and Other Port Related Logic	X	X	X	X		
DMA	X	X		X		13
Wake Up (PM) Context						6
Wake Up Control Register						7
Wake Up Status Registers						8
Function Queue Enable	X	X	X	X		
Function Interrupt and Statistics Registers	X	X	X			
Wake-Up Management Registers	X	X	X	X		2, 9
Memory Configuration Registers	X	X	X	X		2
Flash Request	X	X				12
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 the I210-CS/CL does not support PME from D3cold).
2. 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.SFP_Enable, CTRL_EXT.LINK_MODE, CTRL_EXT.EXT_VLAN* and LED configuration registers are reset on LAN_PWR_GOOD only. Any Flash read resets these fields to the values in the Flash.
   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.
3. The following registers are part of this group:
   a. Max payload size field in PCIe Capability Control register (offset 0xA8).
   b. *Active State Link PM Control* field, *Common Clock Configuration* field and *Extended Synch* field in PCIe Capability Link Control register (Offset 0xB0).
   c. Read *Completion Boundary* in the PCIe Link Control register (Offset 0xB0).
4. 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. GSCL_1/2/3/4
   e. GSCN_0/1/2/3



f. *SW_FW_SYNC* - only part of the bits - see register description for details.

5. The following registers are part of this group:
   a. *MSIX control* register, *MSIX PBA* and *MSIX per vector mask*.
6. 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.
7. 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).
8. The Wake Up Status registers include the following:
   a. Wake Up Status register
   b. Wake Up Packet Length.
   c. Wake Up Packet Memory.
9. 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
10. The other configuration registers include:
    a. General Registers
    b. Interrupt Registers
    c. Receive Registers
    d. Transmit Registers
    e. Statistics Registers
    f. Diagnostic Registers

Of these registers, *MTA[n], VFTA[n], WUPM[n], FTFT[n], FHFT[n], FHFT_EXT[n], TDBAH/TDBAL*, and *RDBAH/RDVAL* registers 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 the I210-CS/CL.

Note: In situations where the device is reset using the software reset *CTRL.RST* or *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. In TBI mode, if the duration is long enough, the link partner might restart the auto-negotiation process by sending break-link (/C/ codes with the configuration register value set to all zeros).

11. 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
12. Includes *EEC.REQ*, *EEC.GNT*, *FLA.REQ* and *FLA.GNT* fields.
13. The following DMA registers are cleared only by LAN_PWR_GOOD, PCIe Reset or *CTRL.DEV_RST*: *DMCTLX, DTPARS, DRPARS* and *DDPARS.*
14. *CTRL.DEV_RST* assertion causes read of function related sections.

4.4 Device and Function Disable

4.4.1 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. The I210-CS/CL provides support for selectively enabling or disabling one or more LAN device(s) in the system.

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. The I210-CS/CL LAN-disable mechanism is implemented in order to be compatible with such a solution.

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

The I210-CS/CL provides a mechanism to disable its LAN port and the PCIe function. When DEV_OFF_N is asserted (driven low) and the *Device Off Enable* Flash bit (in word 0x1E) is set to 1b (refer to Section 6.2.19).

For a LOM design, it might be desirable for the system to provide BIOS-setup capability for selectively enabling or disabling LOM devices. This might allow the end-user more control over system resource-management; avoid conflicts with add-in NIC solutions, etc. The I210-CS/CL provides support for selectively enabling or disabling it.

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 DEV_OFF_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 Flash is read.

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

4.4.3 Disabling PCIe Function Only

The I210-CS/CL also supports disabling just the PCIe function but keeping the LAN port that resides on it fully active. This functionality can be achieved by driving SDP1 pin low and setting Flash en_pin_pcie_func_dis bit (word 0x29) to 1b.

4.4.4 BIOS Handling of Device Disable

4.4.4.1 Sequence for Entering the (Static) Device Off State

1. BIOS recognizes that the entire device should be disabled. The *Device Off Enable* Flash bit in word 0x1E should be set to 1b.
   a. In order to shut down the PHY together with the rest of the device, the *PHY_in_LAN_Disable* Flash bit (refer to Section 6.2.21) should be set to 1b.
2. BIOS asserts the DEV_OFF_N pin (device is on and not in PCIe reset).
3. BIOS issues a PCIe reset
4. PCIe reset sequence ends while the device is already in off state (minimum PCIe reset duration is 100 µs).
5. The BIOS places the *Link in the Electrical IDLE* state (at the other end of the PCIe link) by clearing the *LINK Disable* bit in the Link Control register.
6. BIOS might start with the device enumeration procedure (the I210-CS/CL device function is now invisible).
7. Proceed with normal operation.

4.4.4.2 Sequence for Returning from the (Static) Device Off State

1. Device is in its off state.
2. BIOS de-asserts the DEV_OFF_N pin (while device is off but not while in PCIe reset)
3. BIOS issues a PCIe reset.
4. PCIe reset sequence ends while the device is already in on state (PCIe interface must be operative within 100 ms).
5. BIOS might start with the device enumeration procedure (the I210-CS/CL device function is now visible).
6. Proceed with normal operation.

4.5 Software Initialization and Diagnostics

4.5.1 Introduction

This section discusses general software notes for the I210-CS/CL, 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.

4.5.2 Power Up State

When the I210-CS/CL powers up it reads the iNVM. The iNVM contains sufficient information to bring the link up and configure the I210-CS/CL for 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 detailed timing is given in Section 5.5. The next section gives more details on configuration requirements.

4.5.3 Initialization Sequence

The following sequence of commands is typically issued to the device by the software device driver in order to initialize the I210-CS/CL to normal operation. The major initialization steps are:

- Disable Interrupts - see Interrupts during initialization.
- Issue Global Reset and perform General Configuration - see Global Reset and General Configuration.
- Setup the PHY and the link - see Link Setup Mechanisms and Control/Status Bit Summary.
- Initialize all statistical counters - refer to Section 4.5.8.
- Initialize Receive - refer to Section 4.5.9.
- Initialize Transmit - refer to Section 4.5.10.
- Enable Interrupts - refer to Section 4.5.4.

4.5.4 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 global reset, so a typical driver initialization flow is:
- Disable interrupts
- Issue a Global 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.

4.5.5 Global Reset and General Configuration

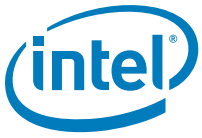
Device initialization typically starts with a global reset that places the device into a known state and enables the software device driver to continue the initialization sequence.

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 PCS layer in SGMII/SerDes mode, 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.

4.5.6 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 4.5.9 for its setting rules.

If DMA coalescing is enabled, to avoid packet loss, the *FCRTC.RTH_Coal* field should also be programmed to a value equal to at least a single maximum packet size below the receive buffer size (a value equal or less than *FCRTH0.RTH* + max size packet).

4.5.7 Link Setup Mechanisms and Control/Status Bit Summary

The *CTRL_EXT.LINK_MODE* value should be set to the desired mode prior to the setting of the other fields in the link setup procedures.

4.5.7.1 MAC/SERDES Link Setup (CTRL_EXT.LINK_MODE = 11b)

Link setup procedures using an external SERDES interface mode:

4.5.7.1.1 Hardware Auto-Negotiation Enabled (PCS_LCTL. AN ENABLE = 1b; CTRL.FRCSPD = 0b; CTRL.FRCDPLX = 0b)

CTRL.FD	Ignored; duplex is set by priority resolution of *PCS_ANDV* and *PCS_LPAB*.
CTRL.SLU	Must be set to 1b by software to enable communications to the SerDes.
CTRL.RFCE	Set by hardware according to auto negotiation resolution[1].
CTRL.TFCE	Set by hardware according to auto negotiation resolution[1].
CTRL.SPEED	Ignored; speed always 1000 Mb/s when using SerDes mode communications.
STATUS.FD	Reflects hardware-negotiated priority resolution.
STATUS.LU	Reflects *PCS_LSTS.AN COMPLETE* (auto-negotiation complete) and link is up.
STATUS.SPEED	Reflects 1000Mb/s speed, reporting fixed value of 10b.
PCS_LCTL.FSD	Must be zero.
PCS_LCTL.Force Flow Control	Must be zero[1].
PCS_LCTL.FSV	Must be set to 10b. Only 1000 Mb/s is supported in SerDes mode.
PCS_LCTL.FDV	Ignored; duplex is set by priority resolution of *PCS_ANDV* and *PCS_LPAB*.
PCS_LCTL.AN TIMEOUT EN	Must be 1b to enable auto-negotiation time-out.
CONNSW.ENRGSRC	Must be 0b on 1000BASE-BX backplane, when source of the signal detect indication is internal. When connected to an optical module and SRDS_[n]_SIG_DET pin is connected to the module, should be 1b.
CTRL.ILOS	If SRDS_[n]_SIG_DET pin connected to optical module, should be set according to optical module polarity.

4.5.7.1.2 Auto-Negotiation Skipped (PCS_LCTL. AN ENABLE = 0b; CTRL.FRCSPD = 1b; CTRL.FRCDPLX = 1b)

CTRL.FD	Must be set to 1b. Only full duplex is supported in SerDes mode.
CTRL.SLU	Must be set to 1b by software to enable communications to the SerDes.

1. If *PCS_LCTL.Force Flow Control* is set, the auto-negotiation result is not reflected in the *CTRL.RFCE* and *CTRL.TFCE* registers. In this case, software must set these fields after reading flow control resolution from PCS registers.

CTRL.RFCE	Must be 0b (No auto-negotiation).
CTRL.TFCE	Must be 0b (No auto-negotiation).
CTRL.SPEED	Must be set to 10b. Only 1000 Mb/s is supported in SerDes mode.
STATUS.FD	Reflects the value written by software to *CTRL.FD*.
STATUS.LU	Reflects whether the PCS is synchronized, qualified with *CTRL.SLU* (set to 1b).
STATUS.SPEED	Reflects 1000 Mb/s speed, reporting fixed value of (10b).
PCS_LCTL.FSD	Must be set to 1b by software to enable communications to the SerDes.
PCS_LCTL.Force Flow Control	Must be set to 1b.
PCS_LCTL.FSV	Must be set to 10b. Only 1000 Mb/s is supported in SerDes mode.
PCS_LCTL.FDV	Must be set to 1b Only full duplex is supported in SerDes mode.
PCS_LCTL.AN TIMEOUT EN	Must be 0b when auto-negotiation is disabled.
CONNSW.ENRGSRC	Must be 0b on 1000BASE-BX backplane, when source of the signal detect indication is internal. When connected to an optical module and SRDS_[n]_SIG_DET pin is connected to the module, should be 1b.
CTRL.ILOS	If SRDS_[n]_SIG_DET pin connected to optical module, should be set according to optical module polarity.

4.5.7.2 MAC/SGMII Link Setup (CTRL_EXT.LINK_MODE = 10b)

Link setup procedures using an external SGMII interface mode:

4.5.7.2.1 Hardware Auto-Negotiation Enabled (PCS_LCTL. AN ENABLE = 1b, CTRL.FRCDPLX = 0b, CTRL.FRCSPD = 0b)

CTRL.FD	Ignored; duplex is set by priority resolution of PCS_ANDV and PCS_LPAB.
CTRL.SLU	Must be set to 1b by software to enable communications to the SerDes.
CTRL.RFCE	Must be programmed by software after reading capabilities from external PHY registers and resolving the desired setting.
CTRL.TFCE	Must be programmed by software after reading capabilities from external PHY registers and resolving the desired setting.
CTRL.SPEED	Ignored; speed setting is established from SGMII's internal indication to the MAC after SGMII PHY has auto-negotiated a successful link-up.
STATUS.FD	Reflects hardware-negotiated priority resolution.
STATUS.LU	Reflects *PCS_LSTS.Link OK*
STATUS.SPEED	Reflects actual speed setting negotiated by the SGMII and indicated to the MAC.
PCS_LCTL.Force Flow Control	Ignored.
PCS_LCTL.FSD	Should be set to zero.
PCS_LCTL.FSV	Ignored; speed is set by priority resolution of *PCS_ANDV* and *PCS_LPAB*.
PCS_LCTL.FDV	Ignored; duplex is set by priority resolution of *PCS_ANDV* and *PCS_LPAB*.
PCS_LCTL.AN TIMEOUT EN	Must be 0b. Auto-negotiation time-out should be disabled in SGMII mode.
CONNSW.ENRGSRC	Must be 0b. In SGMII mode source of the signal detect indication is internal.

4.5.7.3 MAC/1000BASE-KX Link Setup (CTRL_EXT.LINK_MODE = 01b)

4.5.7.3.1 Auto-Negotiation Skipped (PCS_LCTL. AN ENABLE = 0b; CTRL.FRCSPD = 1b; CTRL.FRCDPLX = 1b)

Link setup procedures using an external 1000BASE-KX server backplane interface mode:

CTRL.FD	Must be set to 1b. 1000BASE-KX always in full duplex mode.
CTRL.SLU	Must be set to 1b by software to enable communications to the SerDes.
CTRL.RFCE	Must be 0b (no auto-negotiation).
CTRL.TFCE	Must be 0b (no auto-negotiation).
CTRL.SPEED	Must be set to 10b. Only 1000 Mb/s is supported in 1000BASE-KX mode.
STATUS.FD	Reflects the value written by software to *CTRL.FD*.
STATUS.LU	Reflects whether the PCS is synchronized, qualified with *CTRL.SLU* (set to 1b).
STATUS.SPEED	Reflects 1000Mb/s speed, reporting fixed value of (10b).
PCS_LCTL.FSD	Must be set to 1b by software to enable communications to the 1000BASE-KX SerDes.
PCS_LCTL.Force Flow Control	Must be set to 1b.
PCS_LCTL.FSV	Must be set to 10b. Only 1000 Mb/s is supported in 1000BASE-KX mode.
PCS_LCTL.FDV	Must be set to 1b. Only full duplex is supported in 1000BASE-KX mode.
PCS_LCTL.AN TIMEOUT EN	Must be 0b. Auto-negotiation not supported in 1000BASE-KX mode.
CONNSW.ENRGSRC	Must be 0b. In 1000BASE-KX mode source of the signal detect indication is internal.

4.5.8 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.

4.5.9 Receive Initialization

Program the receive address register(s) per the station address. This can come from the Flash or by any other means (for example, on some machines, this comes from the system PROM not the Flash 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 60 KB:

RXPBSIZE.Rxpbsize + RXPBSIZE.Bmc2ospbsize + TXPBSIZE.Txpb0size + TXPBSIZE.Txpb1size +

TXPBSIZE.Txpb2size + TXPBSIZE.Txpb3size + TXPBSIZE.os2Bmcpbsize <= 60 KB

Program *RCTL* with appropriate values. If initializing it at this stage, 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 needed:

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.
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. The tail should not be bumped before this bit was read as one.
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.

Note: The tail register of the queue (*RDT[n]*) should not be bumped until the queue is enabled.

4.5.9.1 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.

4.5.9.2 Dynamic Enabling and Disabling of Receive Queues

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

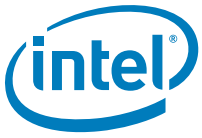
Enabling a queue:

- Follow the per queue initialization sequence described in Section 4.5.9.

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*. The I210-CS/CL stops fetching and writing back descriptors from this queue immediately. The I210-CS/CL 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. The I210-CS/CL 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.

4.5.10 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 4.5.9.

If operation in half duplex mode is expected, program the *TCTL_EXT.COLD* field. For internal PHY mode the default value of 0x42 is acceptable. For SGMII mode, a value reflecting the I210-CS/CL and the PHY SGMII delays should be used. A suggested value for a typical PHY is 0x46 for 10 Mbps and 0x4C for 100 Mb/s.

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* = 1b
  - 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.

Note: The tail register of the queue (*TDT[n]*) should not be bumped until the queue is enabled.

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

4.5.10.1 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.

4.6 Access to Shared Resources

Part of the resources in the I210-CS/CL 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 4.6.1 in order to acquire ownership of this resource and use the flow described in Section 4.6.2 in order to relinquish ownership of this resource.

The shared resources are:

1. Flash.
2. The SerDes 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.

4.6.1 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.

4.6.2 Releasing Ownership Over a Shared Resource

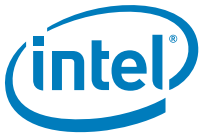
The following flow should be used to release a shared resource:

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.

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



5.0 Power Management

This section describes how power management is implemented in the I210-CS/CL. The I210-CS/CL supports the Advanced Configuration and Power Interface (ACPI) specification as well as Advanced Power Management (APM).

Power management can be disabled via the *power management* bit in the *Initialization Control Word 1* Flash word (see Section 6.2.2), which is loaded during power-up reset. Even when disabled, the power management register set is still present. Power management support is required by the PCIe specification.

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 the I210-CS/CL 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 the I210-CS/CL 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): The I210-CS/CL is completely active and responsive during this D-state. The link can be in either L0 or a low-latency idle state referred to as L0s. Minimizing L0s exit latency is paramount for enabling frequent entry into L0s while facilitating performance needs via a fast exit. A deeper link power state, L1 state, is supported as well.
- D1 and D2: These modes are not supported by the I210-CS/CL.
- 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 the I210-CS/CL with wake-capable logic, or to L3 if no power is delivered to the I210-CS/CL. 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.

Configuring the I210-CS/CL 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. The I210-CS/CL 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 the I210-CS/CL and its link back up, the platform must go through a boot sequence where power, clock, and reset are reapplied appropriately.

5.2 Power States

The I210-CS/CL supports the D0 and D3 architectural power states as described earlier. Internally, the I210-CS/CL 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.

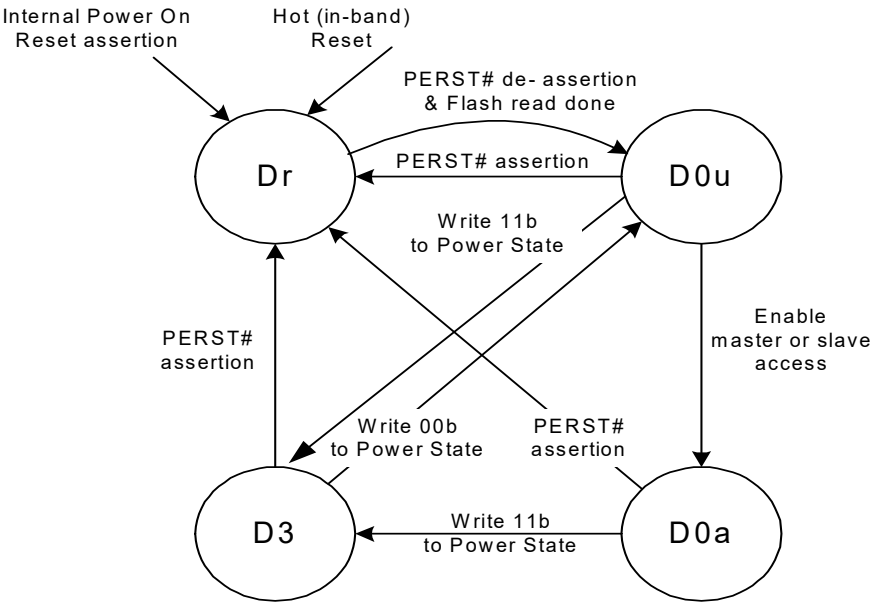


Figure 5-1. Power Management State Diagram

5.2.1 D0 Uninitialized State (D0u)

The D0u state is an architectural low-power state.

When entering D0u, the I210-CS/CL:

- Disables wake up. However APM wake up is enabled (See additional information in Section 5.6.1), if all of the following register bits are set:
  - 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 the I210-CS/CL is cleared, other than sticky bits. State is loaded from the Flash, followed by establishment of the PCIe link. Once this is done, configuration software can access the I210-CS/CL.

On a transition from D3hot state to D0u state, the I210-CS/CL PCI configuration space is not reset (since the *No_Soft_Reset* bit in the *PMCSR* register is set to 1b). However following move to D0a state, the I210-CS/CL requires that the software device driver perform a full re-initialization of the function.

5.2.2 D0active State

Once memory space is enabled, the I210-CS/CL 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. In the I210-CS/CL, 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.
3. Following entry into D0,the software device driver can activate other wake-up filters by writing to the Wake Up Filter Control (*WUFC*) register.

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 8.3.3). The DMA, MAC, and PHY of the appropriate LAN function are also enabled.

5.2.3 D3 State (PCI-PM D3hot)

The I210-CS/CL 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, the I210-CS/CL 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 the I210-CS/CL doesn't clear any bit in the PCIe configuration space. While in D3, the I210-CS/CL 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

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 the I210-CS/CL; 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.

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.

Note: The software device driver can override the *PMCSR.PME_En* bit setting via the *WUC.APMPME* bit.

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 7.21.1, Section 10.8 and Section 10.8.2.3.
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, the system should also set the *Auxiliary (AUX) Power PM Enable* bit in the PCIe Device Control register.

As a response to being programmed into D3 state, the I210-CS/CL transitions its PCIe link into the L1 link state. As part of the transition into L1 state, the I210-CS/CL 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, the I210-CS/CL 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 the I210-CS/CL (and discarded later on D3 exit). Any transmit packets that have not been sent can still be transmitted (assuming the Ethernet link is up).

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 the I210-CS/CL is set to 1b, to indicate that the I210-CS/CL 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 port reset (similar to asserting the *CTRL.RST* bit).

The I210-CS/CL can also be configured via Flash to clear the *No_Soft_Reset* bit in the PMCSR register (see Section 6.2.17). 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, the I210-CS/CL must not issue master accesses. Due to the full-duplex nature of PCIe, and the pipelined design in the I210-CS/CL, 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 the I210-CS/CL function and its software device driver:

- *GIO Master Disable* bit in the Device Control (CTRL) register - When the *GIO Master Disable* bit is set, the I210-CS/CL blocks new master requests by this function. the I210-CS/CL 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 the I210-CS/CL 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 the I210-CS/CL 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 the I210-CS/CL 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, the I210-CS/CL 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). the I210-CS/CL 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)

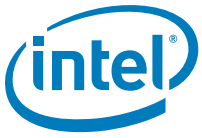
Transition to Dr state is initiated on several occasions:

- 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 D0a 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 D0a to Dr state.
- On transition from a D3 state - The system transitions the I210-CS/CL 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, the I210-CS/CL might enter one of several modes with different levels of functionality and power consumption. The lower-power modes are achieved when the I210-CS/CL is not required to maintain any functionality (see Section 5.2.4.1).



5.2.4.1 Dr Disable Mode

The I210-CS/CL enters a Dr disable mode on transition to D3cold state when it does not need to maintain any functionality. The conditions to enter either state are:

- The I210-CS/CL is in Dr state
- APM WoL (Wake-on-LAN) is inactive
- Proxying is not required (*WUC.PPROXYE* is cleared to 0b).
- ACPI PME is disabled
- The *PHY Power Down Enable* Flash bit is set (word 0xF, bit 6).

Entering Dr disable mode is usually done by asserting PCIe *PE_RST_N*. It might also be possible to enter Dr disable mode by reading the Flash while already in Dr state. The usage model for this later case is on system power up, assuming proxying is not required. Once the I210-CS/CL enters Dr state on power-up, the Flash is read. If the Flash contents determine that the conditions to enter Dr disable mode are met, the I210-CS/CL then enters this mode (assuming that PCIe *PE_RST_N* is still asserted).

The I210-CS/CL exits Dr disable mode when Dr state is exited (See Figure 5-1 for conditions to exit Dr state).

Refer to Section 5.2.6 for details about the static/dynamic device off states built on Dr Disable Mode.

5.2.4.2 Entry to Dr State

Dr entry on platform power-up begins with the assertion of the internal power detection circuit. The Flash is read and determines the I210-CS/CL configuration. If the *APM Enable* bit in the Flash'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 the I210-CS/CL 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.

To reduce power consumption, if PCI-PM PME[1] is enabled, the auto-negotiates to a lower link speed on D0a to Dr transition.

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.

5.2.4.3 Auxiliary Power Usage

The Flash *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

1. ACPI 2.0 specifies that "OSPM will not disable wake events before setting the SLP_EN bit when entering the S5 sleeping state. This provides support for remote management initiatives by enabling Remote Power On (RPO) capability. This is a change from ACPI 1.0 behavior."

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 Flash.

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 Link Disconnect

In any of D0u, D0a, D3, or Dr power states, the I210-CS/CL enters a link-disconnect state if it detects a link-disconnect condition on the Ethernet link. In particular, while in D0 state, software might be able to access any of the I210-CS/CL registers as in a link-connect state.

5.2.6 Device Off States

Note: One single device off mode can be enabled in the Flash at the same time, either Static or Dynamic Device Off mode.

5.2.6.1 (Static) Device Off

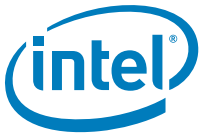
The I210-CS/CL enters a global power-down state when the DEV_OFF_N pin is asserted and the relevant Flash bits were configured as previously described (see Section 4.4.4 for more details on DEV_OFF_N functionality).

5.2.6.2 Dynamic Device Off

The I210-CS/CL enters a global power-down state dynamically, each time all of the following conditions are met:

- The I210-CS/CL *Dynamic Device Off Enable* Flashbit (word 0x1E bit 14) was set (default hardware value is disabled).
- WoL and proxy functionalities are not required

When in this state, the direction of SDP pins is either maintained or the pins are moved to High Impedance according to a setting made in SDP_DDOFF_EN bit in Flash word 0x0A.



5.3 Power Limits by Certain Form Factors

Table 5-1 lists power limitation introduced by different form factors.

Table 5-1. Power Limits by Form-Factor

	Form Factor	
	LOM	PCIe add-in card (10 W slot)
Main	N/A	3 A @ 3.3 V
Auxiliary (aux enabled)	375 mA @ 3.3 V	375 mA @ 3.3 V
Auxiliary (aux disabled)	20 mA @ 3.3 V	20 mA @ 3.3 V

Note: This auxiliary current limit only applies when the primary 3.3 V voltage source is not available (the card is in a low power D3 state).

The I210-CS/CL exceeds the allocated auxiliary power in some configurations.

5.4 Interconnects Power Management

This section describes the power reduction techniques employed by the I210-CS/CL main interconnects.

5.4.1 PCIe Link Power Management

The I210-CS/CL supports all PCIe power management link states:

- L0 state is used in D0u and D0a states.
- The L0s state is used in D0a and D0u states each time link conditions apply.
- The L1 state is 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.

The I210-CS/CL support for active state link power management is reported via the PCIe *Active State Link PM Support* register and is loaded from the Flash.

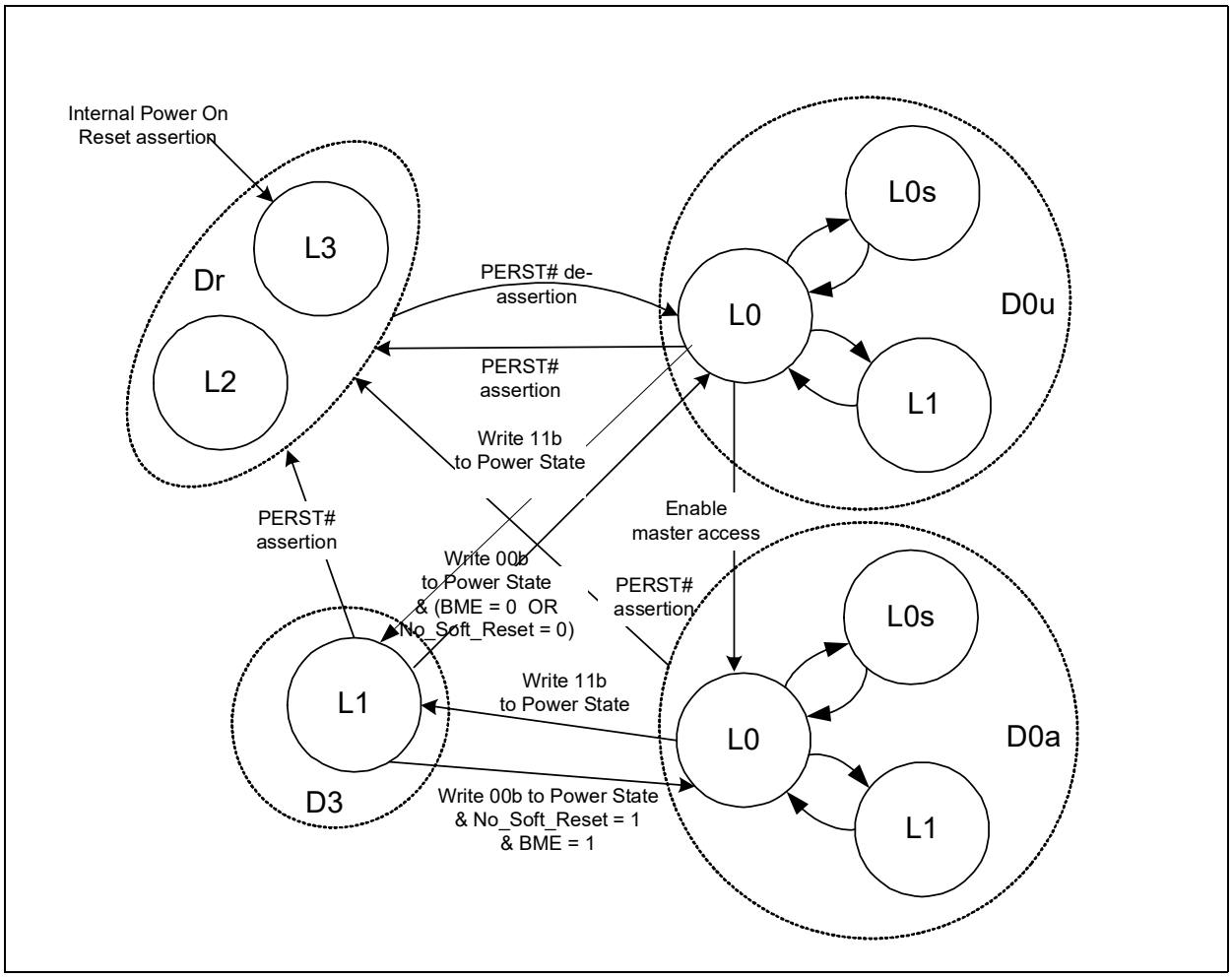
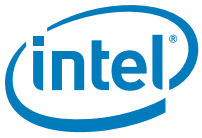


Figure 5-2. Link Power Management State Diagram

While in L0 state, the I210-CS/CL transitions the transmit lane(s) into L0s state once the idle conditions are met for a period of time as follows:

L0s configuration fields are:

- L0s enable - The default value of the *Active State Link PM Control* field in the PCIe Link Control Register is set to 00b (both L0s and L1 disabled). System software might later write a different value into the Link Control register. The default value is loaded on any reset of the PCI configuration registers.
- L0s exit latency (as published in the *L0s Exit Latency* field of the Link Capabilities register) is loaded from Flash. Separate values are loaded when the I210-CS/CL shares the same reference PCIe clock with its partner across the link, and when the I210-CS/CL uses a different reference clock than its partner across the link. The I210-CS/CL reports whether it uses the slot clock configuration through the PCIe Slot Clock Configuration bit loaded from the *Slot_Clock_Cfg* bit in the *PCIe Init Configuration 3* Flash Word.



- L0s Acceptable Latency (as published in the Endpoint L0s Acceptable Latency field of the Device Capabilities Register) is loaded from Flash.

The I210-CS/CL transitions the link into L0s state once the PCIe link has been idle for a period of time defined in the *Latency_To_Enter_L0s* field in the CSR Auto Configuration Power-Up NVM section (see Section 6.3). The I210-CS/CL will then transition the link into L1 state once the PCIe link has been in L0s state for a further period as defined in the *Latency_To_Enter_L1* field in the CSR Auto Configuration Power-Up NVM section.

To comply with the PCIe specification, if the link idle time exceeds the Latency_To_Enter_L0s value defined in the Flash, then the I210-CS/CL enters L0s.

The following Flash 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 (in the L0s state) before the transition into L1 state

5.4.2 SerDes, SGMII and 1000BASE-KX Power Management

The I210-CS/CL SerDes enters a power-down state when none of its clients is enabled and therefore has no need to maintain a link. This can happen in one of the following cases. Note that SerDes and 1000BASE-KX power-down must be enabled through the *SerDes Low Power Enable* bit in Flash word 0x0F.

1. D3/Dr state: SerDes enters a low-power state if the following conditions are met:
   a. The LAN function is in a non-D0 state
   b. APM WOL is inactive
   c. ACPI PME is disabled
   d. The *Dynamic Device Off Enable* Flash bit is set (word 0x1E.14)
2. Device Off: SerDes can be disabled if the DEV_OFF_N pin is asserted. Since SerDes is shared between the LAN function, it might not be desired to power down the SerDes in device disable. The *Lanphy_devoff_pwrdn_confg* NVM field determines whether the SerDes is powered down when the device disable pin is asserted. The default is not to power down.

5.5 Timing of Power-State Transitions

The following sections give detailed timing for the state transitions. In the diagrams, the dotted connecting lines represent the I210-CS/CL requirements, while the solid connecting lines represent the I210-CS/CL guarantees.

The timing diagrams are not to scale. The clocks edges are shown to indicate running clocks only and are not to be used to indicate the actual number of cycles for any operation.

5.5.1 Power Up (Off to Dup to D0u to D0a)

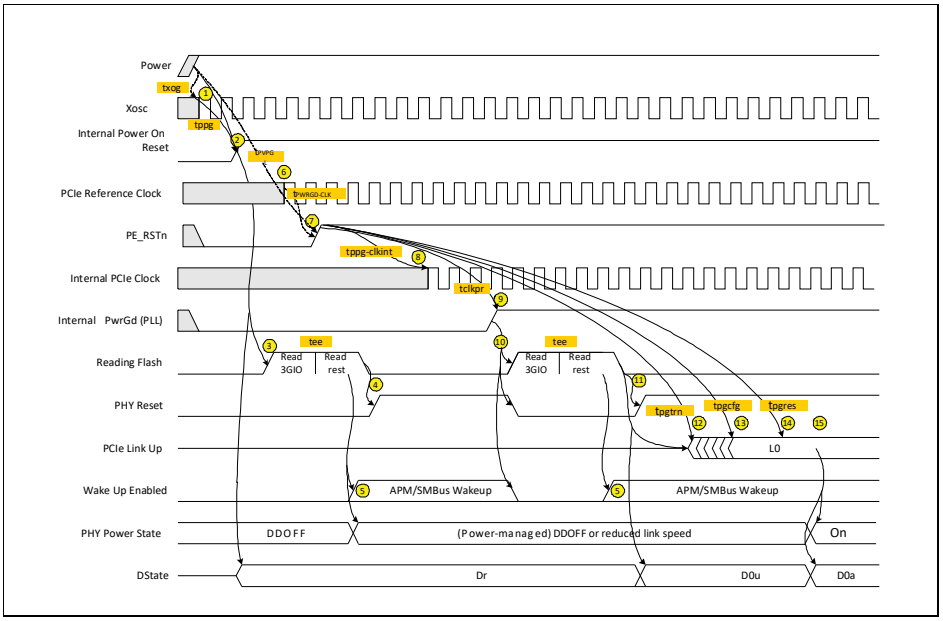


Figure 5-3. Power Up (Off to Dup to D0u to D0a)

Table 5-2. Power Up (Off to Dup to D0u to D0a)

Note	Description
1	Xosc is stable t_{xog} after power is stable.
2	LAN_PWR_GOOD is asserted after all power supplies are good and t_{ppg} after Xosc is stable.
3	A Flash read starts on the rising edge of LAN_PWR_GOOD.
4	After reading the Flash, PHY reset is de-asserted.
5	APM wake-up mode can be enabled based on what is read from the Flash.
6	The PCIe reference clock is valid $t_{PE\_RST-CLK}$ before de-asserting PE_RST_N (according to PCIe specification).
7	PE_RST_N is de-asserted t_{PVPGL} after power is stable (according to PCIe specification).
8	The internal PCIe clock is valid and stable $t_{ppg-clkint}$ from PE_RST_N de-assertion.
9	The PCIe internal PWRGD signal is asserted t_{clkpr} after the external PE_RST_N signal.
10	Asserting internal PCIe PWRGD causes the Flash to be re-read, asserts PHY reset, and disables wake up.
11	After reading the Flash, PHY reset is de-asserted.
12	Link training starts after t_{pgtrn} from PE_RST_N de-assertion.
13	A first PCIe configuration access might arrive after t_{pgcfg} from PE_RST_N de-assertion.
14	A first PCI configuration response can be sent after t_{pgres} from PE_RST_N de-assertion.
15	Writing a 1b to the *Memory Access Enable* bit in the PCI Command Register transitions the I210-CS/CL from D0u to D0 state.

5.5.2 Transition from D0a to D3 and Back Without PE_RST_N

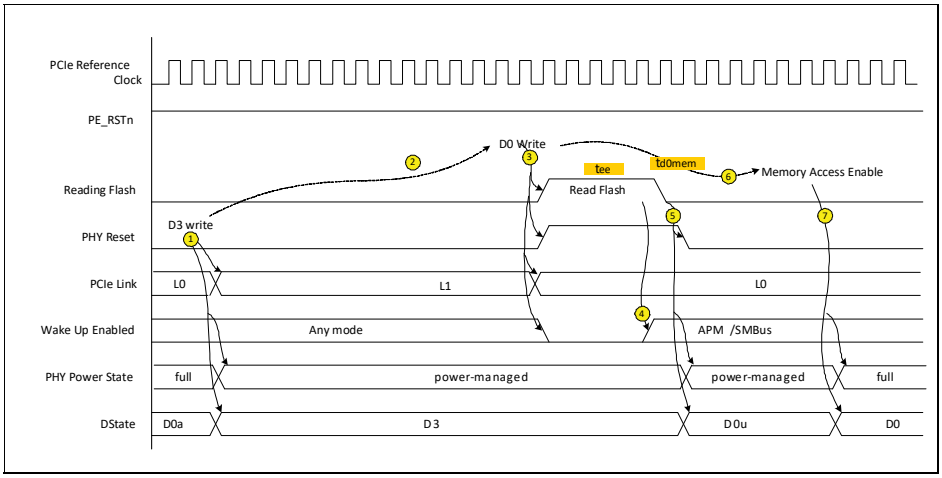


Figure 5-4. Transition from D0a to D3 and Back Without PE_RST_N

Table 5-3. Transition from D0a to D3 and Back Without PE_RST_N

Note	Description
1	Writing 11b to the *Power State* field of the Power Management Control/Status Register (PMCSR) transitions the I210-CS/CL to D3.
2	The system can keep the I210-CS/CL in D3 state for an arbitrary amount of time.
3	To exit D3 state, the system writes 00b to the *Power State* field of the PMCSR.
4	APM wake-up or SMBus mode might be enabled based on what is read in the Flash.
5	After reading the Flash, reset to the PHY is de-asserted. The PHY operates at reduced-speed if APM wake up or SMBus is enabled, else powered-down.
6	The system can delay an arbitrary time before enabling memory access.
7	Writing a 1b to the *Memory Access Enable* bit or to the *I/O Access Enable* bit in the PCI Command Register transitions the I210-CS/CL from D0u to D0 state and returns the PHY to full-power/speed operation.

5.5.3 Transition From D0a to D3 and Back With PE_RST_N

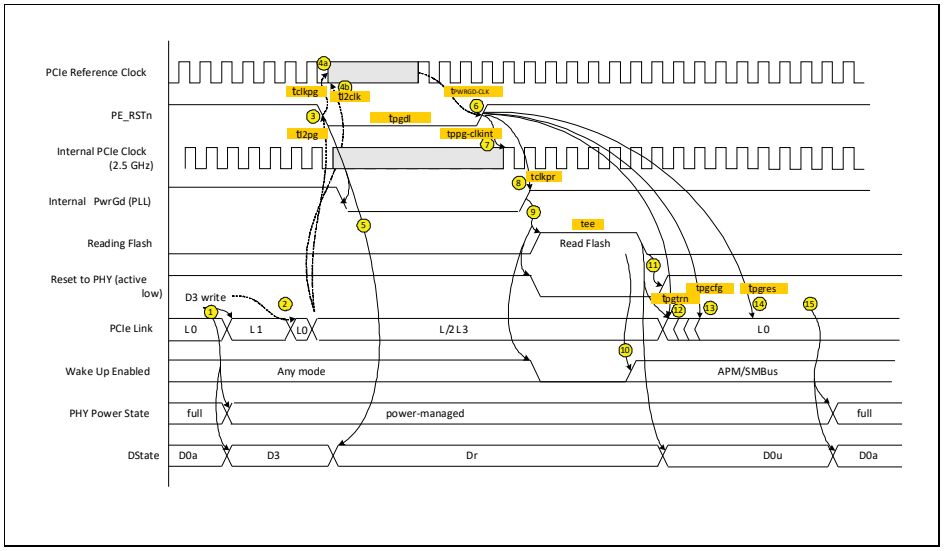


Figure 5-5. Transition From D0a to D3 and Back With PE_RST_N

Table 5-4. Transition From D0a to D3 and Back With PE_RST_N

Note	Description
1	Writing 11b to the *Power State* field of the PMCSR transitions the I210-CS/CL to D3. PCIe link transitions to L1 state.
2	The system can delay an arbitrary amount of time between setting D3 mode and moving the link to a L2 or L3 state.
3	Following link transition, PE_RST_N is asserted.
4	The system must assert PE_RST_N before stopping the PCIe reference clock. It must also wait t_{l2clk} after link transition to L2/L3 before stopping the reference clock.
5	On assertion of PE_RST_N, the I210-CS/CL transitions to Dr state.
6	The system starts the PCIe reference clock $t_{PE\_RST-CLK}$ before de-assertion PE_RST_N.
7	The internal PCIe clock is valid and stable $t_{ppg-clkint}$ from PE_RST_N de-assertion.
8	The PCIe internal PWRGD signal is asserted t_{clkpr} after the external PE_RST_N signal.
9	Asserting internal PCIe PWRGD causes the Flash to be re-read, asserts PHY reset, and disables wake up.
10	APM wake-up mode might be enabled based on what is read from the Flash.
11	After reading the Flash, PHY reset is de-asserted.
12	Link training starts after t_{pgtrn} from PE_RST_N de-assertion.
13	A first PCIe configuration access might arrive after t_{pgcfg} from PE_RST_N de-assertion.
14	A first PCI configuration response can be sent after t_{pgres} from PE_RST_N de-assertion.
15	Writing a 1b to the *Memory Access Enable* bit in the PCI Command Register transitions the I210-CS/CL from D0u to D0 state.

5.5.4 Transition From D0a to Dr and Back Without Transition to D3

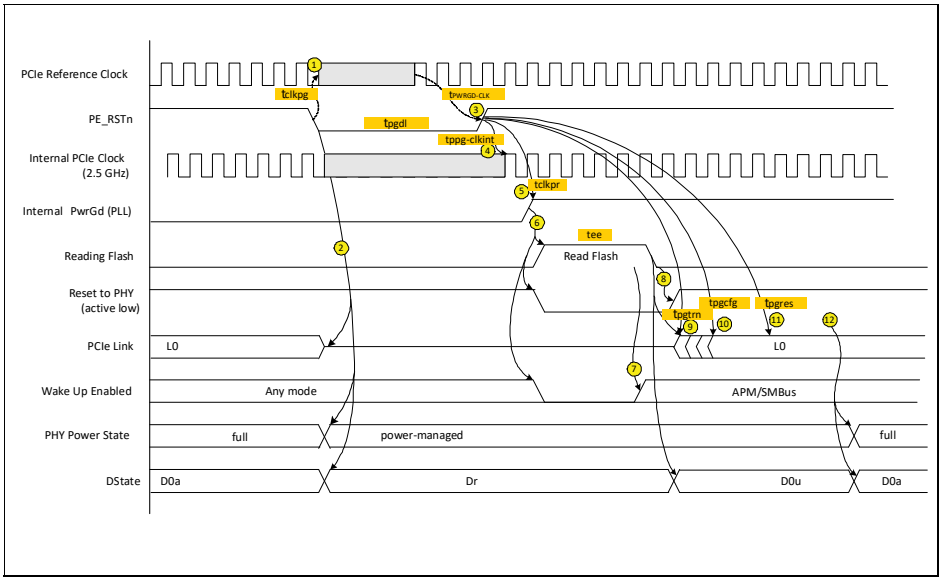


Figure 5-6. Transition From D0a to Dr and Back Without Transition to D3

Table 5-5. Transition From D0a to Dr and Back Without Transition to D3

Note	Description
1	The system must assert PE_RST_N before stopping the PCIe reference clock. It must also wait t_{l2clk} after link transition to L2/L3 before stopping the reference clock.
2	On assertion of PE_RST_N, the I210-CS/CL transitions to Dr state and the PCIe link transition to electrical idle.
3	The system starts the PCIe reference clock $t_{PE\_RST-CLK}$ before de-assertion PE_RST_N.
4	The internal PCIe clock is valid and stable $t_{ppg-clkint}$ from PE_RST_N de-assertion.
5	The PCIe internal PWRGD signal is asserted t_{clkpr} after the external PE_RST_N signal.
6	Asserting internal PCIe PWRGD causes the Flash to be re-read, asserts PHY reset, and disables wake up.
7	APM wake-up mode might be enabled based on what is read from the Flash.
8	After reading the Flash, PHY reset is de-asserted.
9	Link training starts after t_{pgtrn} from PE_RST_N de-assertion.
10	A first PCIe configuration access might arrive after t_{pgcfg} from PE_RST_N de-assertion.
11	A first PCI configuration response can be sent after t_{pgres} from PE_RST_N de-assertion.
12	Writing a 1b to the *Memory Access Enable* bit in the PCI Command Register transitions the I210-CS/CL from D0u to D0 state.

5.5.5 Timing Requirements

The I210-CS/CL requires the following start-up and power state transitions.

Parameter	Description	Min.	Max.	Notes
t_{xog}	Xosc stable from power stable		56 ms	
$t_{PE\_RST-CLK}$	PCIe clock valid to PCIe power good	100 µs	-	According to PCIe spec.
t_{PVPGL}	Power rails stable to PCIe PE_RST active	100 ms	-	According to PCIe spec.
T_{pgcfg}	External PE_RST signal to first configuration cycle.	100 ms		According to PCIe spec.
t_{d0mem}	Device programmed from D3h to D0 state to next device access	10 ms		According to PCI power management spec.
t_{l2pg}	L2 link transition to PE_RST de-assertion	0 ns		According to PCIe spec.
t_{l2clk}	L2 link transition to removal of PCIe reference clock	100 ns		According to PCIe spec.
T_{clkpg}	PE_RST de-assertion to removal of PCIe reference clock	0 ns		According to PCIe spec.
T_{pgdl}	PE_RST de-assertion time	100 µs		According to PCIe spec.

5.5.6 Timing Guarantees

The I210-CS/CL guarantees the following start-up and power state transition related timing parameters.

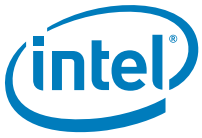
Parameter	Description	Min.	Max.	Notes
t_{xog}	Xosc stable from power stable		56 msec	
t_{ppg}	Internal power good delay from valid power rail		45 msec	
t_{ee}	NVM read duration		20 msec	
$t_{ppg-clkint}$	PCIe* PE_RST to internal PLL lock	-	5 ms	
t_{clkpr}	Internal PCIe PWGD from external PCIe PE_RST		50 µs	
t_{pgtrn}	PCIe PE_RST to start of link training		20 ms	According to PCIe spec.
t_{pgres}	External PE_RST to response to first configuration cycle		1 s	According to PCIe spec.

5.6 Wake Up

The I210-CS/CL 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, the I210-CS/CL 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.



5.6.1 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. The I210-CS/CL now uses (if configured) an in-band PM_PME message for this functionality.

On power up, the I210-CS/CL reads the *APM Enable* bits from the Flash *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, the I210-CS/CL checks all incoming packets for Magic packets.

Once the I210-CS/CL 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 128 bytes of the packet in the Wake Up Packet Memory (WUPM) register.
- 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.

The I210-CS/CL 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 Flash 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 the I210-CS/CL 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.

5.6.2 ACPI Power Management Wake Up

The I210-CS/CL 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).

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) register.
- 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). It can also set the *WUFC.LNKC* bit to cause wake up on link status change.
- Once the I210-CS/CL wakes the system, the software device driver needs to clear the WUS and WUFC 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, the I210-CS/CL 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, the I210-CS/CL:

- 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 128 bytes of the packet in the Wakeup Packet Memory (WUPM) register.
- Sets one or more bits in the Wake Up Status (WUS) register. Note that the I210-CS/CL 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, the I210-CS/CL ignores any subsequent wake-up packets until the software device driver clears all of the received bits in the Wake Up Status (WUS) 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.

Note: A wake on link change is not supported when configured to SerDes or 1000BASE-KX mode.

5.6.3 Wake-Up and Proxying Filters

The I210-CS/CL 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.

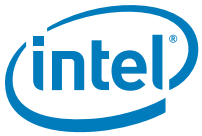


Table 5-6. ARP Packet Structure and Processing

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	
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*.

5.7 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, the I210-CS/CL 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, the I210-CS/CL supports up to two IPv6 Multicast-Address-Specific Multicast Listener Discovery (MLD) queries (either MLDv1 or MLDv2). In addition, the I210-CS/CL 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 the I210-CS/CL.

— 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 the I210-CS/CL (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 the I210-CS/CL 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 the I210-CS/CL functionality and enablement for mDNS Proxy Offload. See section 5.7.3

In addition to the D3 low power functionality, by setting *D0_PROXY* bit to 1b, the I210-CS/CL 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.

5.7.1 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 for proxying and then program the necessary data to the IPv4/v6 Address Table (IP4AT, IP6AT) and the Flexible Host Filter Table (FHFT) 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 4.6.1 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 10.8.2.4.2.2).
7. Verify that the *HICR.En* bit (See Section 7.21.2) 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 10.8.2.4.2. 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 4.6.2.
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 the I210-CS/CL 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, the I210-CS/CL 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, the I210-CS/CL:

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: The I210-CS/CL 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 10.8.2.4.2.2).

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, the I210-CS/CL should respond to the request after less than 60 Seconds.

5.7.2 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 for proxying and then program the necessary data to the IPv4/v6 Address Table (IP4AT, IP6AT) and the Flexible Host Filter Table (FHFT) registers.
4. Take ownership of the management host interface semaphore (*SW_FW_SYNC.SW_MNG_SM* register bit) using the flow defined in Section 4.6.1 to send protocol offload information to firmware.
5. Verify that the *HICR.En* bit 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 10.8.2.4.2. 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 4.6.2.
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, the I210-CS/CL 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 the I210-CS/CL:

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.

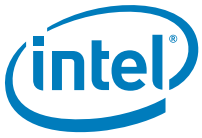
Note: The I210-CS/CL sets more than one bit in the PROXYS register if a packet matches more than one filter.

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.

5.7.3 mDNS Proxy Offload

The I210-CS/CL 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 5.7.1).
  - The I210-CS/CL 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 3-5 and to Section 6.8.11, 6.8.12. 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 10.8.2.4.2.5 to activate the mDNS proxy.

When activated, the mDNS proxy must act as a responsible mDNS responder.

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 5-7. mDNS Offload Configuration

Frame Type	Address/Protocol	Why Needed	I210-CS/CL 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 6.8.11 and Section 6.8.12). Refer to the mDNS Proxy SAS document for the exact structure of the mDNS data section to be stored in the Flash.

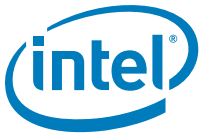
Note: IP fragments are not supported for mDNS proxy offload, filtering of higher layers (ICMP, TCP/ UDP ports etc.) is not supported on IP fragments.

Note: The mDNS proxy offload will ignore any IPv4 options and silently drop all IPv6 packets with extensions.

5.8 DMA Coalescing

The I210-CS/CL supports DMA coalescing to enable synchronizing port activity and optimize power management of memory, CPU and RC internal circuitry. When conditions to enter DMA coalescing operating mode as defined in Section 5.8.2 exist, the I210-CS/CL:

- Stops initiation of any activity on the PCIe link.
- Data received from the Ethernet link is buffered in internal receive buffer.
- When executing DMA coalescing, once the internal Tx buffer is empty, the internal Rx buffer watermark for transmission of XOFF flow control packets on the network is defined by the *FCRTC.RTH_Coal* threshold field.



The I210-CS/CL exits DMA coalescing once the conditions defined in Section 5.8.3, to exit DMA coalescing, exist.

5.8.1 DMA Coalescing Activation

To activate DMA coalescing functionality software driver should program the following fields:

1. *DMACR.DMACTHR* field to set the receive threshold that causes move out of DMA coalescing operating mode. Receive watermark programmed should take into account latency tolerance reported and L1 to L0 latency to avoid receive buffer overflow when DMA coalescing is enabled. A minimum of 70 us equivalent is recommended.
2. *DMCTXTH.DMCTTHR* field to set transmit threshold that causes move out of DMA coalescing operating mode. Transmit watermark programmed should take into account latency tolerance reported and L1 to L0 latency to enable transmission of back-to-back packets when DMA Coalescing is enabled.
3. *DMACR.DMACWT* field that defines a maximum timeout value for:
   a. A receive packet to be stored in the internal receive buffer before the I210-CS/CL moves a packet to host memory.
   b. Time to delay an interrupt that is not defined as an immediate interrupt in the IMIR[n], IMIREXT[n] or IMIRVP registers, when other conditions specified in Section 5.8.3 to exit DMA coalescing do not exist.
4. *DMCTLX.DCFLUSH_DIS* to define if pending descriptor write-back flush and pending interrupt flush should occur before entry into DMA coalescing state.
   — When *DMCTLX.DCFLUSH_DIS* is set to 1b, any pending interrupts or descriptor write-back operations do not cause the I210-CS/CL to move out of a DMA coalescing state.
5. *FCRTC.RTH_Coal* field that defines a flow control receive high watermark for sending flow control packets. The I210-CS/CL uses the *FCRTC.RTH_Coal* threshold when:
   — Flow control is enabled by setting the *CTRL.TFCE* bit.
   — The I210-CS/CL is in DMA coalescing mode.
   — Internal transmit buffer is empty.
6. *SRRCTL[n].DMACQ_Dis* bit to define high priority queues. When a received packet is forwarded to a queue with the *SRRCTL[n].DMACQ_Dis* bit set, the I210-CS/CL moves immediately out of DMA coalescing mode and executes a DMA operation to store the packet in host memory.
7. *DMACR.DMAC_EN* bit should be set to 1b to enable activation of DMA coalescing operating mode.
8. DMCMNGTH.DMCMNGTHR field to set the threshold for the management buffer that causes move out of DMA coalescing operating mode.

Notes:

1. The values of *DMACR.DMACTHR* and *FCRTC.RTH_Coal* should be set so that XOFF packet generation is avoided. In DMA coalescing mode, when the transmit buffer is empty, the XOFF flow control threshold (*FCRTC.RTH_Coal*) value can be increased by the maximum jumbo frame size compared to normal operation, where the high threshold is set by the FCRTH0 register.
2. When entering DMA coalescing mode, the value written in the FCRTH0 register is used to generate XOFF flow control frames until the internal transmit buffer is empty. Once the internal transmit buffer is empty the value written in the *FCRTC.RTH_Coal* field is used as a watermark for generation of XOFF flow control frames.
3. The I210-CS/CL transitions the link into L0s state once the PCIe link has been idle for a period of time defined in the *Latency_To_Enter_L0s* field in the CSR Auto Configuration Power-Up Flash section (see Section 6.3). The I210-CS/CL will then transition the link into L1 state once the PCIe link has been in L0s state for a further period as defined in the *Latency_To_Enter_L1* field in the CSR Auto Configuration Power-Up NVM section.

5.8.2 Entering DMA Coalescing Operating Mode

Enabling DMA coalescing operation by setting the *DMACR.DMAC_EN* bit to 1b. Power saving is achieved since it increases the duration of these idle intervals. The Power Management Unit (PMU) on the platform can use these idle intervals to reduce system power.

5.8.2.1 Entering DMA Coalescing

The I210-CS/CL enters DMA coalescing when all of the following conditions exist:

1. DMA coalescing is enabled (*DMACR.DMAC_EN* = 1b).
2. Internal receive buffers (host and management if enabled) are empty.
3. There are no pending DMA operations.
4. None of the conditions defined in Section 5.8.3.1 to move out of DMA coalescing exist.

Before entering the DMA coalescing power saving mode, if the *DMCTLX.DCFLUSH_DIS* bit is programmed to 0b, the I210-CS/CL:

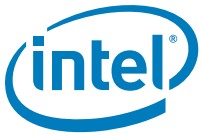
- Flushes all pending interrupts that were delayed due to the Interrupt Throttling (ITR) mechanism.
- The I210-CS/CL flushes all pending receive descriptor and transmit descriptor write backs and pre-fetch available receive descriptors and transmit descriptors to the internal cache.

5.8.3 Conditions to Exit DMA Coalescing

5.8.3.1 Exiting DMA Coalescing

When the I210-CS/CL is in DMA coalescing operating mode, DMA coalescing mode is exited when one of the following events occurs:

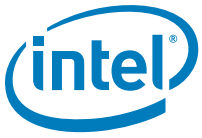
1. Empty space in the internal transmit buffer is above the value defined in the *DMCTXTH.DMCTTHR* field and available transmit descriptors exist.
2. A high priority packet was received (see Section 6.3.6 for a definition of high priority packets). A high priority packet is a packet that generates an immediate interrupt, as defined in the IMIR[n], IMIREXT[n] or IMIRVP registers.
3. A received packet destined to a high priority queue (*SRRCTL[n].DMACQ_Dis* =1b) was detected.
4. DMA coalescing watchdog timer defined in the *DMACR.DMACWT* field expires as a result of the following occurrences not being serviced for the duration defined in the *DMACR.DMACWT* field:
   — An Rx packet was received in the internal buffer.
   — An interrupt is pending.
   — A descriptor write-back is pending.
   — On-chip transmit tail pointer was updated.
5. Received data rate detected is lower than defined in the *DMCRTRH.UTRESH* field.
6. DMA coalescing is disabled (*DMACR.DMAC_EN* = 0b).
7. Software initiates a move out of DMA coalescing by writing 1b to the *DMACR.EXIT_DC* self-clearing bit.
8. MC to OS traffic if the *DMACR.DC_BMC2OSW_EN* bit is programmed to 0b.
9. Management indications are enabled through DMCTLX.EN_MNG_IND and the amount of data buffered in the management buffer exceeds DMCMNGTH.DMCMNGTHR.



Notes:

1. Even when conditions for DMA coalescing do not exist, the I210-CS/CL continues to be in a low power PCIe link state (L0s or L1) if there is no requirement for PCIe access.
2. If a PCIe PME wake message needs to be sent, the PCIe link moves from an L1 low power state to L0 to send the message but DMA remains in the DMA coalescing state.
3. Pending interrupts or pending descriptor write-back operations do not cause the I210-CS/CL to move out of the DMA coalescing state.

NOTE: This page intentionally left blank.



6.0 Inline Functions

6.1 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 in queue assignment is to verify that the packet is destined to the port. This is done by a set of L2 filters as described in Section 6.1.3.

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 6.1.1.

6.1.1 L2 Packet Filtering

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. This section describes how host filtering is done.

As shown in Figure 6-1, host filtering has two stages:

1. Packets are filtered by L2 filters (MAC address, unicast/multicast/broadcast). See Section 6.1.1.1 for details.
2. Packets are then filtered by VLAN if a VLAN tag is present. See Section 6.1.1.2 for details.

A packet is not forwarded to the host if any of the following takes place:

1. The packet does not pass MAC address filters as described later in this section.
2. The packet does not pass VLAN filtering as described later in this section.

A packet that passes receive filtering as previously described 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 6.1.1.3), 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 the I210-CS/CL (even bad packets).

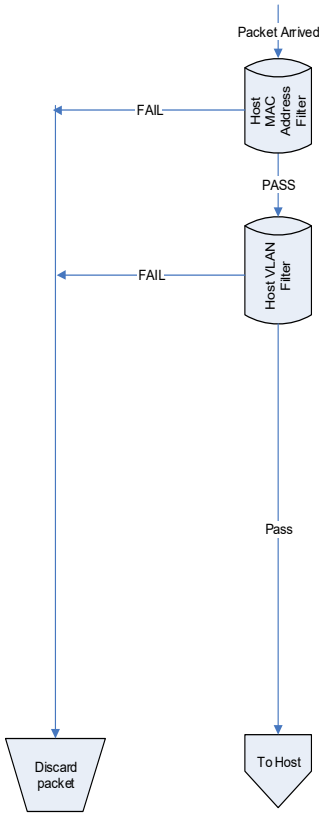


Figure 6-1. I210-CS/CL Receive Filtering Flow Chart

6.1.1.1 MAC Address Filtering

Figure 6-2 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.
2. It is a multicast packet and promiscuous multicast filtering is enabled.
3. It is a unicast packet and it matches one of the unicast MAC filters.
4. It is a multicast packet and it matches one of the multicast filters.
5. It is a broadcast packet and Broadcast Accept Mode (*RCTL.BAM*) is enabled.

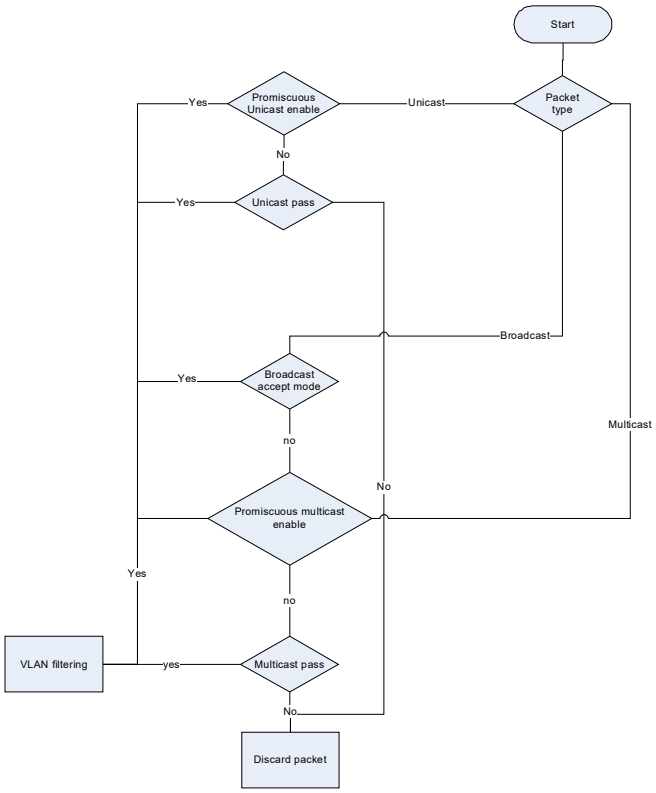


Figure 6-2. Host MAC Address Receive Filtering Flow Chart

6.1.1.1.1 Unicast Filter

The entire MAC address is checked against the 16 host unicast addresses. The 16 host unicast addresses are controlled by the host interface The destination address of an incoming packet must exactly match one of the pre-configured host address filters. These addresses can be unicast or multicast. Those filters are configured through *RAL*, and *RAH* registers.

Promiscuous Unicast — Receive all unicasts. Promiscuous unicast mode in the *RCTL* register can be set/cleared only through the host interface. This mode is usually used when the I210-CS/CL is used as a sniffer.

6.1.1.1.2 Multicast Filter (Inexact)

A 12-bit portion of incoming packet multicast address must exactly match Multicast Filter Address (MFA) in order to pass multicast filtering. This means that 12 bits out of 48 bits of the destination address are used and can be selected by the *MO* field of RCTL (Section 7.10.1). The 12 bits extracted from the Multicast Destination address are used as an address for a bit in the Multicast Table Array (MTA). If the value of the bit selected in the MTA table is 1b, the packet is sent to the host (See Section 7.10.15). These entries can be configured only by the host interface and cannot be controlled by the MC. Packets received according to this mode have the *PIF* bit in the descriptor set to indicate imperfect filtering that should be validated by the software device driver.

Promiscuous Multicast — Receive all multicast. Promiscuous multicast mode can be set/cleared in the RCTL register only through the host interface and it is usually used when the I210-CS/CL is used as a sniffer.

Note: When the promiscuous bit is set and a multicast packet is received, the *PIF* bit of the packet status is not set.

6.1.1.2 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 6-3 shows the VLAN filtering flow.

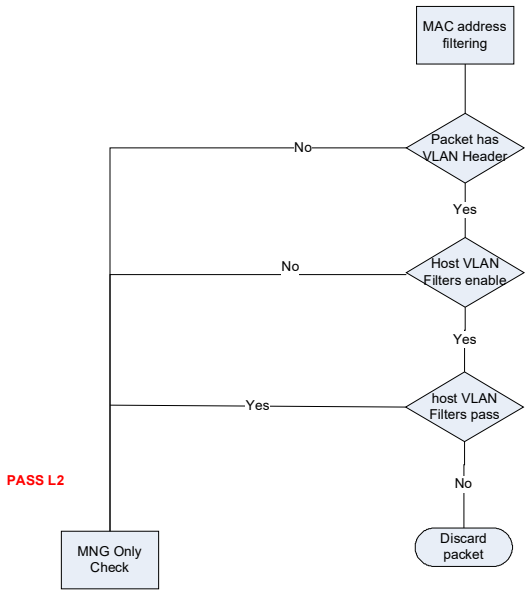
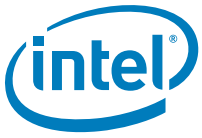


Figure 6-3. I210-CS/CL VLAN Filtering



6.1.1.3 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).

6.1.2 Receive Queues Assignment

The following filter mechanisms determines the destination of a receive packet. These are described briefly in this section and in full details in separate sections:

- 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 6.1.2.7 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 6.1.2.3 for mode details. The I210-CS/CL 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. The I210-CS/CL has 8 such filters. See Section 6.1.2.4 for details.
- TCP SYN filters — The I210-CS/CL 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. The I210-CS/CL has one such filter. See Section 6.1.2.6 for more details.
- Flex Filters - These filters can be either used as WoL filters when the I210-CS/CL 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. The I210-CS/CL has 8 such filters. See Section 6.1.2.5 for details.
- VLAN priority filters — These filters identify packets by their L2 VLAN priority and assign them to receive queues. See Section 6.1.2.7 for mode details. The I210-CS/CL 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 6.1.2.8 for mode details. The I210-CS/CL incorporates 16 MAC address filters.

A received packet is allocated to a queue as described in the following sections.

6.1.2.1 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 6-7 shows the receive queue assignment flow.

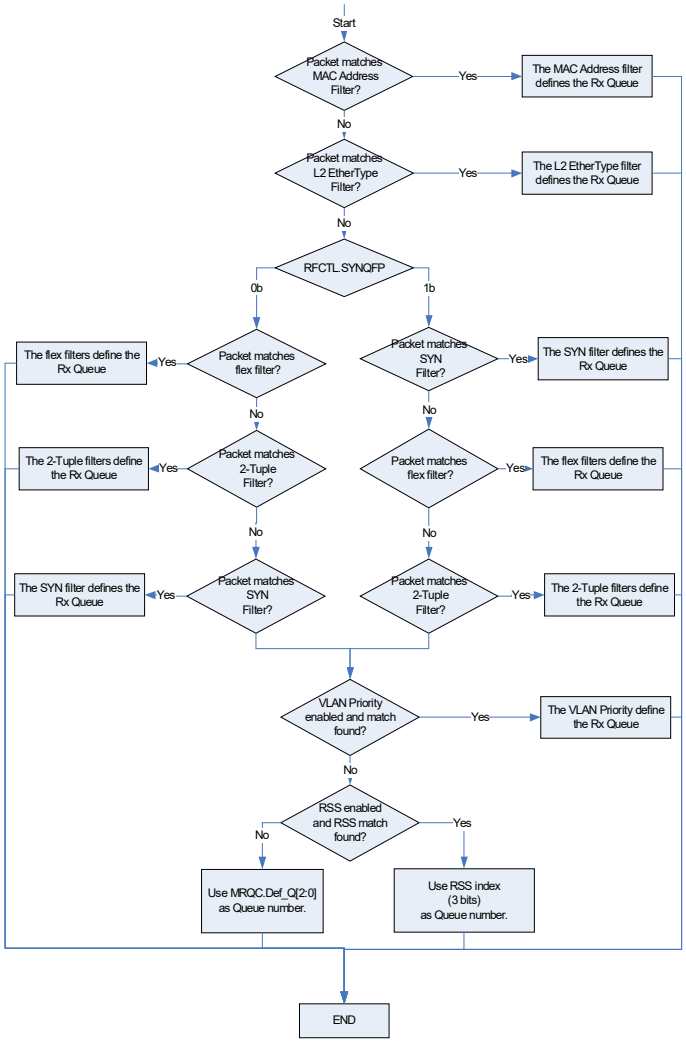


Figure 6-4. Receive Queuing Flow

6.1.2.2 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
- *RXCTL* — Rx DCA Control
- *SRRCTL* — Split and Replication Receive Control
- *PSRTYPE* — Packet Split Receive Type

6.1.2.3 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.

The I210-CS/CL incorporates 4 Ether-type filters.

The *Packet Type* field in the Rx descriptor captures the filter number that matched the L2 Ether-type. See Section 6.1.4.2 for decoding of the *Packet Type* field.

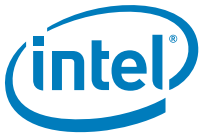
The Ether-type filters are configured via the ETQF register as follows:

- 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 6.1.2.1 on the impact and order of ETQF on the I210-CS/CL 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.



6.1.2.4 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.

The I210-CS/CL incorporates 8 such filters.

The 2-tuple filters are configured via the *TTQF* (See Section 7.11.3), *IMIR* (See Section 7.11.1) and *IMIR_EXT* (See Section 7.11.2) 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.

Notes:

- If more than one 2-tuple filter with the same priority is matched by 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.

6.1.2.5 Flex Filters

The I210-CS/CL supports a total of 8 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*, See Section 7.20.18 and Section 7.20.19). These 8 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 See* Section 7.20.2) register or Proxying Filter Control (*PROXYFC* see Section 7.20.6) 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. In D3 mode these filters can be used for WoL or proxying as described in Section 5.7.

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*, *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.
- Filter enable - Set *WUFC.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 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.

6.1.2.6 SYN Packet Filters

The I210-CS/CL 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.

6.1.2.7 VLAN Priority Filters

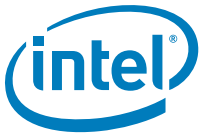
The I210-CS/CL can forward packets according to their VLAN priority to separate queues. The I210-CS/CL supports the configuration of the destination queue per VLAN priority.

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.

6.1.2.8 VLAN Tag Filters

The I210-CS/CL can forward packets according to their VLAN tag to separate queues. The I210-CS/CL supports the configuration of the destination queue per VLAN tag.



VLAN tag filters are configured via the VLANTAGQF registers as follows:

- VLAN tag value - The VLAN tag value to be filtered
- Queue En — Enables forwarding of packets for each filtered VLAN tag to a specific queue.
- Rx Queue field — Contains the destination queue for each filtered VLAN tag packet.

6.1.2.9 MAC Address Filters

The I210-CS/CL can forward packets according to their MAC address to separate queues. The I210-CS/CL 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.

6.1.2.10 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.

The I210-CS/CL uses RSS as one ingredient in its packet assignment policy (the others are the various filters for Qav). The RSS output is a RSS index. The I210-CS/CL'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, the I210-CS/CL 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 6-6 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. The I210-CS/CL supports a single hash function, as defined by Microsoft* RSS. The I210-CS/CL 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.

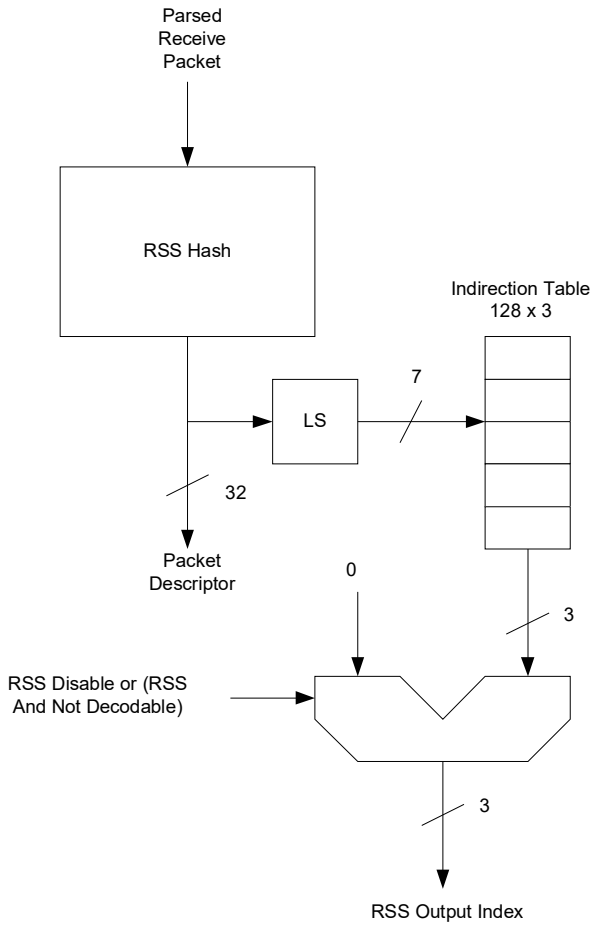


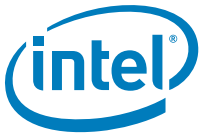
Figure 6-5. RSS Block Diagram

6.1.2.10.1 RSS Hash Function

Section 6.1.2.10.1 provides a verification suite used to validate that the hash function is computed according to Microsoft* nomenclature.

The I210-CS/CL hash function follows Microsoft* definition. A single hash function is defined with several variations for the following cases:

- TcpIPv4 — The I210-CS/CL 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 — The I210-CS/CL parses the packet to identify an IPv4 packet. If the packet is not an IPv4 packet, RSS is not done for the packet.
- TcpIPv6 — The I210-CS/CL 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 — The I210-CS/CL 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 — The I210-CS/CL 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 — The I210-CS/CL 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 — The I210-CS/CL parses the packet to identify a packet with UDP over IPv4.
- UdpIPV6 — The I210-CS/CL parses the packet to identify a packet with UDP over IPv6.
- UdpIPV6Ex — The I210-CS/CL 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.

6.1.2.10.1.1 Hash for IPv4 with TCP

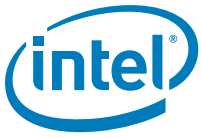
Concatenate SourceAddress, DestinationAddress, SourcePort, DestinationPort into one single byte-array, preserving the order in which they occurred in the packet:

```
Input[12] = @12-15, @16-19, @20-21, @22-23.
Result = ComputeHash(Input, 12);
```

6.1.2.10.1.2 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);
```



6.1.2.10.1.3 Hash for IPv4 without TCP

Concatenate SourceAddress and DestinationAddress into one single byte-array.

```
Input[8] = @12-15, @16-19
Result = ComputeHash(Input, 8)
```

6.1.2.10.1.4 Hash for IPv6 with TCP

Similar to above:

```
Input[36] = @8-23, @24-39, @40-41, @42-43
Result = ComputeHash(Input, 36)
```

6.1.2.10.1.5 Hash for IPv6 with UDP

Similar to above:

```
Input[36] = @8-23, @24-39, @40-41, @42-43
Result = ComputeHash(Input, 36)
```

6.1.2.10.1.6 Hash for IPv6 without TCP

```
Input[32] = @8-23, @24-39
Result = ComputeHash(Input, 32)
```

6.1.2.10.2 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

Note: In RSS only mode, all 3 bits are used. In VMDq mode RSS is not supported.

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.

6.1.2.10.3 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
```

6.1.2.10.3.1 IPv4

Table 6-1. IPv4

Destination Address/Port	Source Address/Port	IPv4 Only	IPv4 With TCP
161.142.100.80:1766	66.9.149.187:2794	0x323e8fc2	0x51ccc178
65.69.140.83:4739	199.92.111.2:14230	0xd718262a	0xc626b0ea
12.22.207.184:38024	24.19.198.95:12898	0xd2d0a5de	0x5c2b394a
209.142.163.6:2217	38.27.205.30:48228	0x82989176	0xafc7327f
202.188.127.2:1303	153.39.163.191:44251	0x5d1809c5	0x10e828a2

6.1.2.10.3.2 IPv6

The IPv6 address tuples are only for verification purposes and might not make sense as a tuple.

Table 6-2. 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

6.1.2.10.4 Association Through MAC Address

Each of the 16 MAC address filters can be associated with a VM. The *POOLSEL* field in the Receive Address High (*RAH*) register determines the target VM. Packets that do not match any of the MAC filters (such as promiscuous) are assigned with the default VM as defined in the *VT_CTL.DEF_PL* field.

Software can program different values to the MAC filters (any bits in *RAH* or *RAL*) at any time. The I210-CS/CL would respond to the change on a packet boundary but does not guarantee the change to take place at some precise time.

6.1.3 Receive Data Storage

6.1.3.1 Host Buffers

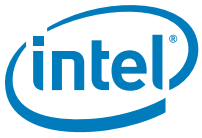
Each descriptor points to a one or more memory buffers that are designated by the software device driver to store packet data.

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.

The I210-CS/CL 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.

6.1.3.2 On-Chip Receive Buffer

The I210-CS/CL allocates by default a 36 KB on-chip packet buffer. The buffer is used to store packets until they are forwarded to the host. Actual on-chip receive buffer allocated can be controlled the *RXPBSIZE register*.

6.1.3.3 On-chip Descriptor Buffers

The I210-CS/CL 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 6.1.4.3 and Section 6.1.4.4.

6.1.4 Receive Descriptors

6.1.4.1 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, the I210-CS/CL uses the legacy Receive descriptor listed in Table 6-3. The shaded areas indicate fields that are modified by hardware upon packet reception (so-called descriptor write-back).

Table 6-3. Legacy Receive Descriptor (RDESC) Layout

	63 48	47 40	39 32	31 16	15 0
0	Buffer Address [63:0]				
8	VLAN Tag	Errors	Status	Fragment Checksum	Length

After receiving a packet for the I210-CS/CL, hardware stores the packet data into the indicated buffer and writes the length, packet checksum, status, errors, and status fields.

Packet Buffer Address (64) - Physical address of the packet buffer.

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 6.1.7.2

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 6-4 for the layout of the *Status* field. Error status information is shown in Figure 6-8.

Table 6-4. 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 6-5 lists the meaning of these bits:

Table 6-5. 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 6.4.

IPCS (IPv4 Checksum), L4CS (L4 Checksum), and UDPCS (UDP Checksum)

The meaning of these bits are listed in Table 6-6:

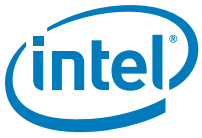


Table 6-6. IPCS, L4CS, and UDPCS

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.
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.

Refer to Table 6-18 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 the I210-CS/CL 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 packets passing only the Multicast Vector (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 6-7 for a definition of the possible errors and their bit positions.

Table 6-7. 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 6-7 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 3.6.3.3.2) 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 6-8. VLAN Tag Field Layout (for 802.1Q Packet)

15 13	12	11 0
PRI	CFI	VLAN

6.1.4.2 Advanced Receive Descriptors

6.1.4.2.1 Advanced Receive Descriptors (RDESC) - Read Format

Table 6-9 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 6-10.

Table 6-9. RDESC Descriptor Read Format

	63 1	0
0	Packet Buffer Address [63:1]	A0/NSE
8	Header Buffer Address [63:1]	DD

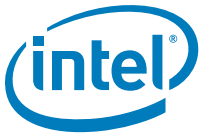
Packet Buffer Address (64) - Physical address of the packet buffer. The lowest bit is either A0 (LSB of address) or NSE (No-Snoop Enable), depending on bit *RXCTL.RXdataWriteNSEn* of the relevant queue. See Section 7.13.1.

Header Buffer Address (64) - Physical address of the header buffer. The lowest bit is *DD*.

Note: The I210-CS/CL 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, the I210-CS/CL 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*.

6.1.4.2.2 Advanced Receive Descriptors (RDESC) - Write-back Format

When the I210-CS/CL writes back the descriptors, it uses the descriptor format shown in Table 6-10.

Note: *SRRCTL[n]. DESCTYPE* must be set to a value other than 000b for the I210-CS/CL to write back the special descriptors.

Table 6-10. 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 6-11. 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

The I210-CS/CL 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
- 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 validL2 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 ETQF Valid (bit 11) - L2 ETQF field in Packet Type is valid. Higher layer indications (bits 7:0) can still be set.

The 11 LSB bits of the packet type reports the packet type identified by the hardware as follows:

Table 6-12. 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 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.

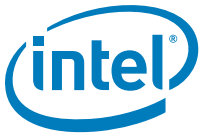
RSV(22):

Reserved.

HDR_LEN (10) - The length (bytes) of the header as parsed by the I210-CS/CL. 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 the I210-CS/CL 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 6.1.7.

Packet types supported by the header split and header replication are listed in Appendix A.1. Other packet types are posted sequentially in the host packet buffer. Each line in Table 6-13 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 6.1.5 while the packet types for this functionality are enabled by the *PSRTYPE[n]* registers (Section 7.10.3).

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.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 the I210-CS/CL 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 6.1.7.2. 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 6.1.7.2. 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 6.1.2.7. 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 6-13 lists the extended status word in the last descriptor of a packet (*EOP* is set). Table 6-14 lists the extended status word in any descriptor but the last one of a packet (*EOP* is cleared).

Table 6-13. Receive Status (RDESC.STATUS) Layout of the Last Descriptor

19	18	17	16	15	14	13	12	11	10
MC	Rsv	Rsv	TS	TSIP	Reserved		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 6-14. 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 10.4.TS (16) - Time Stamped Packet (Time Sync). The *Time Stamp* bit is set to indicate that the device recognized a Time Sync packet and time stamped it in the RXSTMPL/H time stamp registers (See Section 6.8.2.3).

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 6.1.7.

Reserved (2, 8, 14:13, 17, 18) - Reserved at zero.

PIF (7), IPCS(6), UDPCS(4), VP(3), EOP (1), DD (0) - These bits are described in the legacy descriptor format in Section 6.1.4.

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 6.4.5.

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 6.1.7.2. 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.

Extended Error (12)

Table 6-15 and the text that follows describes the possible errors reported by hardware.

Table 6-15. Receive Errors (RDESC.ERRORS) Layout

11	10	9	8 7	6 4	3	2 0
RXE	IPE	L4E	Reserved	Reserved	HBO	Reserved

RXE (bit 11)

RXE is described in the legacy descriptor format in Section 6.1.4.

IPE (bit 10)

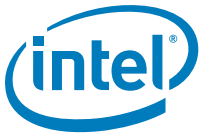
The IPE error indication is described in the legacy descriptor format in Section 6.1.4.

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.

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 6.1.7.

VLAN Tag (16)

These bits are described in the legacy descriptor format in Section 6.1.4.

6.1.4.3 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, the I210-CS/CL 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.

6.1.4.4 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 the I210-CS/CL 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.

6.1.4.5 Receive Descriptor Ring Structure

Figure 6-7 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.

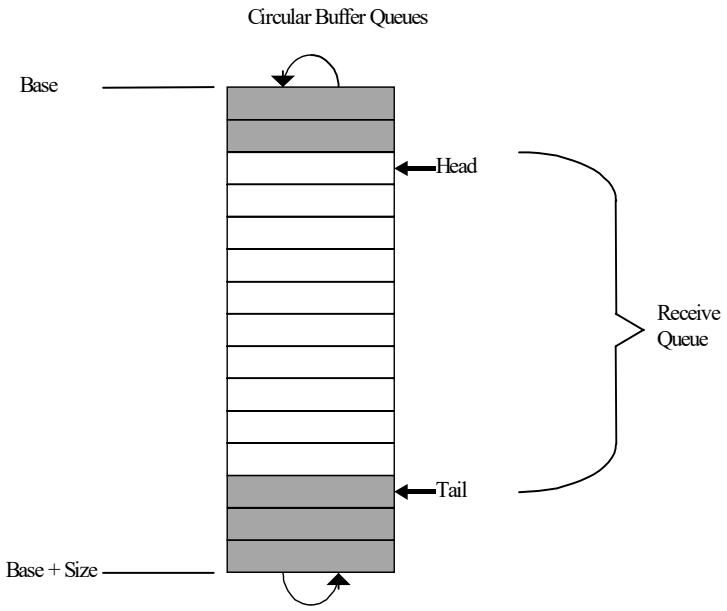


Figure 6-6. Receive Descriptor Ring Structure

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 6-7 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.

6.1.4.5.1 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, the I210-CS/CL 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.

6.1.5 Header Splitting and Replication

6.1.5.1 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 the DCA feature enabled (see Section 7.13) or in conjunction with a software-prefetch.

The packet (header and payload) is stored in memory through a (optionally) non-snoop transaction. Later, a transaction moves the payload from the software device driver buffer to application memory or it is moved using a normal memory copy operation.

The I210-CS/CL 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.

6.1.5.2 Description

In Figure 6-7 and Figure 6-8, the header splitting and header replication modes are shown.

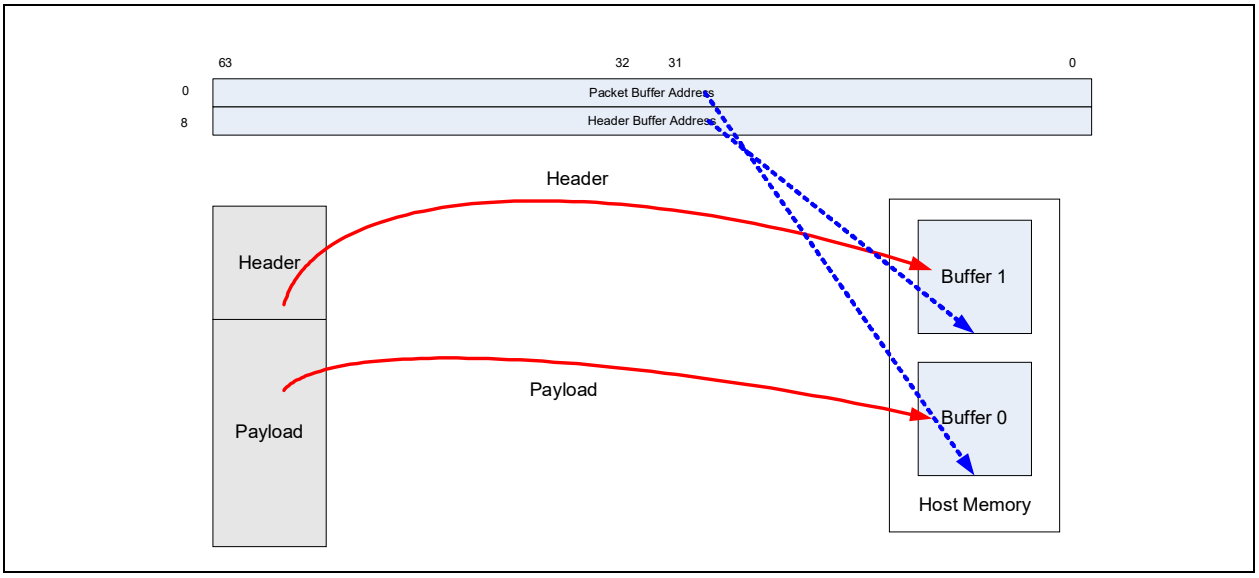


Figure 6-7. Header Splitting

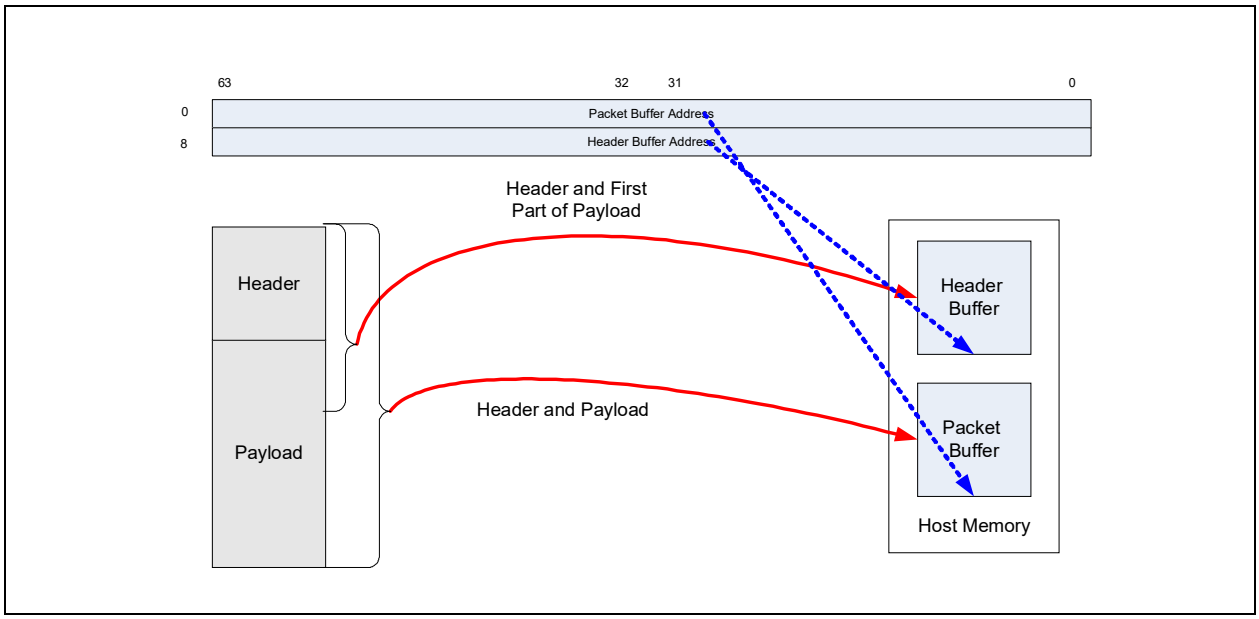


Figure 6-8. Header Replication

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.

The I210-CS/CL 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 A.1 for details on the possible headers type supported).

Table 6-16 lists the behavior of the I210-CS/CL in the different modes.

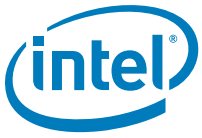


Table 6-16. I210-CS/CL Split/Replicated Header Behavior

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
Replicate	1. Header can't be decoded	0b[3]	0b	Min(Packet length, *BSIZEPACKET*)	N/A	(Header + Payload) (partial[5]) ⇨ Header buffer Header + Payload ⇨ Packet buffer
	2. Packet length <= *BSIZEHEADER*	1b[3]	0b	Min(Packet length, *BSIZEPACKET*)	Header size	Header + Payload ⇨ Header buffer Header + Payload ⇨ Packet buffer
	3. Packet length > *BSIZEHEADER*	1b[3]	0b/1b[4]	Min(Packet length, *BSIZEPACKET*)	Header size	Header + Payload (partial[5]) ⇨ Header buffer Header + Payload ⇨ Packet buffer
Replicate Large Packet only	1. Header can't be decoded	0b[3]	0b	Min(Packet length, *BSIZEPACKET*)	N/A	(Header + Payload) (partial[5]) ⇨ Header buffer Header + Payload ⇨ Packet buffer
	2. Packet length <= *BSIZEHEADER*	1b[3]	0b	Packet length	Header size	Header + Payload ⇨ Header buffer
	2. Packet length > *BSIZEHEADER*	1b[3]	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. 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.
4. HBO is 1b if the header size is bigger than *BSIZEHEADER* and zero otherwise.
5. 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 A.1 describes the details of the split/replicate conditions for different types of headers according to the settings of the *PSRTYPE* register values.

6.1.6 Receive Packet Timestamp in Buffer

The I210-CS/CL supports adding an optional tailored header before the MAC header of the packet in the receive buffer. The 64 MSB bits of 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 (See Section 6.8.3.1 for further information on *SYSTIML/H* operation). The 64 LSB bits of the tailored header are reserved.

The timestamp information is placed in Host order (Little Endian) format as listed in Table 6-17.

Table 6-17. Timestamp Layout in Buffer

0 3	4 7	8 11	12 15	16...
Reserved (0x0)	Reserved (0x0)	SYSTIML	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
- 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 6-69 within Section 6.9.1. Meeting these cases the packet is reported as follow:

- Place a 64 bit timestamp, indicating the time a packet was received by the MAC, 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.
- Update the *RDESC.Packet Type* field in the last receive descriptor. 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.

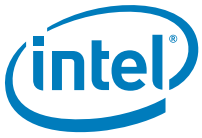
While the receive path is disabled, the Timestamp in Buffer mode can be disabled by clearing RXPBSIZE.cfg_ts_en flag and issuing a Port Software Reset event (CTRL.RST).

6.1.7 Receive Packet Checksum and SCTP CRC Offloading

The I210-CS/CL supports the off loading of four receive checksum calculations: packet checksum, fragment payload checksum, the IPv4 header checksum, and the TCP/UDP checksum. In addition, SCTP CRC32 calculation is supported as described in Section 6.1.7.3

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 6.1.7.2 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 the I210-CS/CL. If *RXCSUM.IPOFLD* is set to 1b, the I210-CS/CL 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, the I210-CS/CL 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).

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 6-18. 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
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

Table 6-18. Supported Receive Checksum Capabilities

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/ UDP Checksum Calculation	Hardware SCTP CRC Calculation
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.

6.1.7.1 Filters Details

Table 6-18 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:

6.1.7.1.1 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.

6.1.7.1.2 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*.

6.1.7.1.3 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.

6.1.7.1.4 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).

6.1.7.1.5 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 6-19:

Table 6-19. 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 2 3
Next Header Type	Length	

Table 6-20 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 the I210-CS/CL so their details are not covered here.

Table 6-20. 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 - the I210-CS/CL cannot offload packets with this header type.

Note that the I210-CS/CL hardware acceleration does not support all IPv6 extension header types (refer to Table 6-18).

6.1.7.1.6 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.

6.1.7.2 Receive UDP Fragmentation Checksum

The I210-CS/CL might provide receive fragmented UDP checksum offload. The I210-CS/CL 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 6-21 lists the exact description of the checksum calculation.

Table 6-21 also lists the outcome descriptor fields for the following incoming packets types:

Table 6-21. 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 6.1.7 for supported packet formats.

6.1.7.3 SCTP Offload

If a receive packet is identified as SCTP, the I210-CS/CL 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:

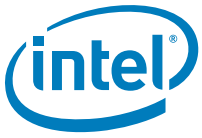


Table 6-22. SCTP Header

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			

6.2 Transmit Functionality

6.2.1 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 the I210-CS/CL 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 6.2.4.

In addition, the I210-CS/CL supports SCTP offloading for transmit requests. See section Section 6.2.5.3 for details about SCTP.

Table 1-10 provides a high level description of all data/control transformation steps needed for sending Ethernet packets to the line.

6.2.1.1 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.

6.2.1.2 On-Chip Transmit Buffers

The I210-CS/CL 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.*

6.2.1.3 On-Chip descriptor Buffers

The I210-CS/CL 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 6.2.2.5 and Section 6.2.2.6.

6.2.1.4 Transmit Contexts

The I210-CS/CL 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. The I210-CS/CL 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 the I210-CS/CL.

The I210-CS/CL 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.

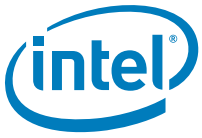
Contexts can be initialized with a transmit context descriptor and then used for a series of related transmit data descriptors. The context, for example, defines the checksum and offload capabilities for a given type of TCP/IP flow. All packets of this type can be sent using this context.

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.

6.2.2 Transmit Descriptors

The I210-CS/CL supports legacy descriptors and the I210-CS/CL advanced descriptors.

Legacy descriptors are intended to support legacy drivers to enable fast platform power up and to facilitate debug.

In addition, the I210-CS/CL 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.

6.2.2.1 Legacy Transmit Descriptor Format

Legacy 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 6-23. Note that the address and length must be supplied by software. Also note that bits in the command byte are optional, as is the CSO field.

Table 6-23. 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	Reserved	Reserved	STA	CMD	CSO	Length

Table 6-24. 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	Reserved	Reserved	STA	CMD	CSO	Length

Note: For frames that span multiple descriptors, the *VLAN, CSO, CMD.VLE, CMD.IC,* and *CMD.IFCS* are valid only in the first descriptors and are ignored in the subsequent ones.

6.2.2.1.1 Buffer Address (64)

Physical address of a data buffer in host memory that contains a portion of a transmit packet.

6.2.2.1.2 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 can change, based on the value written to the *DMA TX Max Allowable packet size* (*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.

6.2.2.1.3 Checksum Offset and Start - CSO

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.

CSO is in a unit of bytes and must be in the range of data provided to the I210-CS/CL in the descriptors. For short packets that are not padded by software, 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. Checksum calculation is not done if CSO is out of range. This occurs if (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: UDP checksum calculation is not supported by the legacy descriptors. When using legacy descriptors the I210-CS/CL 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.

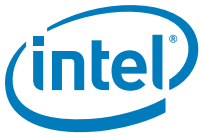
Hardware adds the checksum to the field at the offset indicated by the *CSO* field. A value of zero corresponds to the first byte in the packet.

Table 6-25. Transmit Command (TDESC.CMD) Layout

7	6	5	4	3	2	1	0
RSV	VLE	DEXT	Rsv	RS	IC	IFCS	EOP

6.2.2.1.4 Command Byte - CMD

The CMD byte stores the applicable command and has the fields shown in Figure 6-25.



- RSV (bit 7) - Reserved
- VLE (bit 6) - VLAN Insertion Enable (See Table 6-26).
- 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 6-26. 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 6-27 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.

Note: VLE, IFCS, CSO, and IC must be set correctly only in the first descriptor of each packet. In previous silicon generations, some of these bits were required to be set in the last descriptor of a packet.

6.2.2.1.5 Status – STA

Table 6-27. Transmit Status (TDESC.STA) Layout

3	2	1	0
Reserved			DD

6.2.2.1.6 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).

6.2.2.1.7 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 6-28. 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).

6.2.2.2 Advanced Transmit Context Descriptor

Table 6-29. Transmit Context Descriptor (TDESC) Layout - (Type = 0010b)

	63 57	56 32	31 16	15 9	8 0
0	Reserved	LaunchTime	VLAN	MACLEN	IPLEN

	63 48	47 40	39	38 36	35 30	29	28 24	23 20	19 9	8 0
8	MSS	L4LEN	RSV[1]	IDX	Reserved	DEXT	RSV[1]	DTYP	TUCMD	Reserved

1. RSV - Reserved

6.2.2.2.1 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.

6.2.2.2.2 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 6-30 lists the value of *MACLEN* in the different cases.

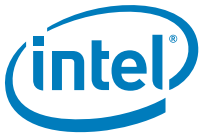


Table 6-30. MACLEN Values

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
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 6-28.

Note: The VLAN tag is sent in network order.

6.2.2.2.3 LaunchTime (25)

The LaunchTime field is only used in Qav mode when a queue is configured as SR queue. The LaunchTime value is used to

1. calculate the fetch time - this time defines the time to fetch the packet from the host to the packet buffer, and
2. define the launch time - the time to transmit a packet from the packet buffer.

The LaunchTime is a 25 bit field defined in 32 nsec units (Launch time = LaunchTime * 32). The Launch time is compared against the SYSTIML register ignoring the SYSTIMH value. It means that the Launch time is defined relative to the fraction of a second of the SYSTIM.

- The Launch time is defined as expired if it is smaller or equal to the SYSTIML.
- The Launch time is defined as greater than SYSTIML if it is in the range of [SYSTIML … SYSTIML + 0.5 sec).
- It is defined as smaller than SYSTIML if it is in the range of [SYSTIML - 0.5 sec … SYSTIML).

Notes: The operations SYSTIML + 0.5 sec and SYSTIML - 0.5 sec are modulo 1 sec.

For meaningful operation, the Launch time should never be set to larger value than SYSTIML + 0.5sec. Otherwise, the Launch time might be misinterpreted.

The LaunchTime parameter is a relative time to the LaunchOffset parameter in the LAUNCH_OS0 register. So, the actual Launch time equals to 32 * (LaunchOffset + LaunchTime).

6.2.2.2.4 TUCMD (11)

Table 6-31. 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

6.2.2.2.5 DTYP(4)

Always 0010b for this type of descriptor.

6.2.2.2.6 DEXT(1)

Descriptor Extension (1b for advanced mode).

6.2.2.2.7 IDX (3)

Index into the hardware context table where this context is stored. In the I210-CS/CL the 2 available register context sets per queue are accessed using the LSB bit and the two MSB bits are reserved and should always be 0.

Note: In Qav mode for the SR queues a valid context descriptor should be placed ahead of any timed packet pointed by a data descriptor and the IDX field is ignored.

6.2.2.2.8 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.

6.2.2.2.9 MSS (16)

Controls the Maximum Segment Size (MSS). 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:

Total length is equal to:

MACLEN + 4(if VLE set) + IPLEN + L4LEN + MSS

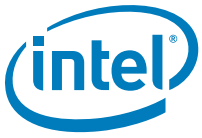
The one exception is the last packet of a TCP/UDP segmentation, which is typically shorter.

MSS is ignored when *DCMD.TSE* is not set.

Note: The headers lengths must meet the following:

MACLEN + IPLEN + L4LEN <= 512

Note: The MSS value should be larger than 0 and the maximum MSS value should not exceed 9216 bytes (9 KB) length.



The context descriptor requires valid data only in the fields used by the specific offload options. Table 6-32 lists the required valid fields according to the different offload options.

Table 6-32. 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
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.

6.2.2.3 Advanced Transmit Data Descriptor

Table 6-33. Advanced Transmit Data Descriptor (TDESD) Layout - (Type = 0011b)

0	Address[63:0]									
8	PAYLEN	POPTS	RSV[1]	IDX	STA	DCMD	DTYP	MAC	RSV[1]	DTALEN
	63 46	**45 40**	**39**	**38 36**	**35 32**	**31 24**	**23 20**	**19 18**	**17 16**	**15 0**

1. RSV - Reserved

Table 6-34. 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.

6.2.2.3.1 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 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.

- *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

6.2.2.3.2 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.

6.2.2.3.3 MAC (2)

Table 6-35. Transmit Data (TDESD.MAC) Layout

1	0
2STEP_1588	1STEP_1588

- 1STEP_1588 (bit 1) - Sample IEEE1588 Timestamp and post it in the transmitted packet at the offset defined by the *1588_Offset* field in the *TSYNCTXCTL* register.
- 2STEP_1588 (bit 1) - Sample IEEE1588 Timestamp at packet transmission in the TXSTMP registers.

Note: The two flags 1STEP_1588 and 2STEP_1588 are mutually.

6.2.2.3.4 DTYP (4)

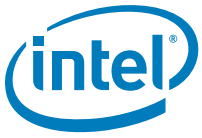
0011b is the value for this descriptor type.

6.2.2.3.5 DCMD (8)

Table 6-36. 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.

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.

6.2.2.3.6 STA (4)

- Rsv (bits 2-3) - Reserved
- TS_STAT (bit 1) - DMA Timestamp is provided in the *DMA Time Stamp* field. It is enabled by the 1588_STAT_EN flag in the TQAVCTRL register.
- DD (bit 0) - Descriptor Done

6.2.2.3.7 IDX (3)

Index into the hardware context table to indicate which context should 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 6-32 for details on type of transmit packet offloads that require a context reference.

6.2.2.3.8 POPTS (6)

Table 6-37. Transmit Data (TDESD.POPTS) Layout

5 2	1	0
Reserved	TXSM	IXSM

- Reserved (bits 5:2)
- TXSM (bit 1) - Insert L4 Checksum
- IXSM (bit 0) - Insert IP Checksum

TXSM, when set to 1b, L4 checksum must be inserted. In this case, TUCMD.L4T in the context descriptor indicates whether the checksum is TCP, UDP, or SCTP.

When *DCMD.TSE* in TDESD is set, *TXSM* must be set to 1b.

If this bit is set, the packet should at least contain a TCP header.

IXSM, when set to 1b, indicates that IP checksum must be inserted. For IPv6 packets this bit must be cleared.

If the *DCMD.TSE* bit is set in data descriptor, and *TUCMD.IPV4* is set in context descriptor, *POPTS.IXSM* must be set to 1b as well.

If this bit is set, the packet should at least contain an IP header.

6.2.2.3.9 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.

6.2.2.4 Transmit Descriptor Ring Structure

The transmit descriptor ring structure is shown in Figure 6-9. 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.

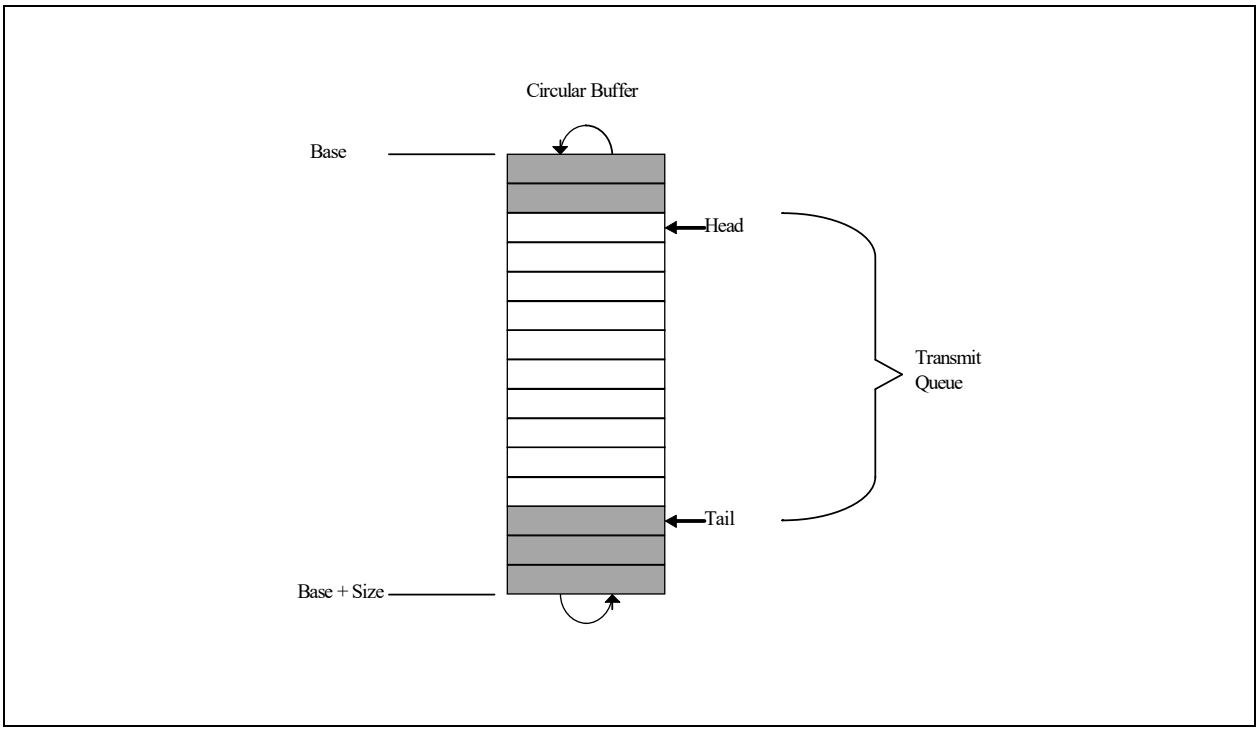


Figure 6-9. Transmit Descriptor Ring Structure

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 the I210-CS/CL 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, the I210-CS/CL 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 6.2.3 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.

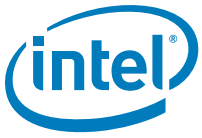
All the registers controlling the descriptor rings behavior should be set before transmit is enabled, apart from the tail registers which are used during the regular flow of data.

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 the I210-CS/CL to the address pointed by the *TDWBAL* and *TDWBAH* registers (see Section 6.2.3 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.

6.2.2.5 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 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, highest indexed queue is served first and so forth, down to the lowest 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 internal descriptor 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, the I210-CS/CL 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: The I210-CS/CL NEVER fetches descriptors beyond the descriptor tail pointer.

6.2.2.6 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.

Any descriptor write-back includes the full 16 bytes of the descriptor.

Since the benefit of delaying and then bursting transmit descriptor write-backs is small at best, it is likely that the threshold is left at the default value (0x0) to force immediate write-back of transmit descriptors with their *RS* bit set and to preserve backward compatibility.

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.

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.

6.2.2.7 Transmit Descriptor Ring Structure

- *TXDCTL[n].WTHRESH* > 0x0 and *TXDCTL[n].WTHRESH* descriptors have accumulated.

For the first condition, write-backs are immediate. This is the default operation and is backward compatible with previous Intel Ethernet controllers.

The other two conditions are only valid if descriptor bursting is enabled (Section 7.12.15). In the second condition, the *EITR* counter is used to force timely write-back of descriptors. The first packet after timer initialization starts the timer. Timer expiration flushes any accumulated descriptors and sets an interrupt event (*TXDW*).

For the last condition, if *TXDCTL[n].WTHRESH* descriptors are ready for write-back, the write-back is performed.

An additional mode in which transmit descriptors are not written back at all and the head pointer of the descriptor ring is written instead as described in Section 6.2.3.

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.

6.2.3 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. In order to improve the performance of this feature, the software device driver might program DCA registers to configure which CPU is processing each TX queue to allow pre-fetching of the head write back value from the right cache.

6.2.3.1 Description

The head counter is reflected in a memory location that is allocated by software, for each queue.

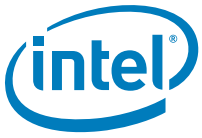
Head write back occurs if *TDWBAL[n].Head_WB_En* is set for this queue and the *RS* bit is set in the Tx descriptor, following corresponding data upload into packet buffer. If the head write-back feature is enabled, the I210-CS/CL ignores *TXDCTL[n].WTHRESH* and takes in account only descriptors with 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 7.12.16 and Section 7.12.17.

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 7.13.2.

The I210-CS/CL 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 the I210-CS/CL).

Note: Software should program TDWBAL[n], TDWBAH[n] registers when queue is disabled (TXDCTL[n].Enable = 0).

6.2.4 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.

The I210-CS/CL 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.

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.

The offloading of these mechanisms from the software device driver to the I210-CS/CL saves significant CPU cycles. Note that the software device driver shares the additional tasks to support these options.

6.2.4.1 Assumptions

The following assumptions apply to the TCP segmentation implementation in the I210-CS/CL:

- 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.

6.2.4.2 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 the I210-CS/CL 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).

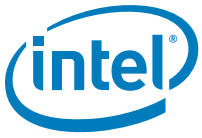
6.2.4.2.1 TCP Segmentation Data Fetch Control

To perform TCP Segmentation in the I210-CS/CL, 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. The I210-CS/CL 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.

6.2.4.2.2 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 6-38.

Table 6-38. 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.

6.2.4.3 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.

6.2.4.4 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. The I210-CS/CL partitions the data packet into standard Ethernet frames prior to transmission according to the requested MSS. The I210-CS/CL supports calculating the Ethernet, IP, TCP, and UDP headers, including checksum, on a frame-by-frame basis.

Table 6-39. 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 6-40. TCP/IP or UDP/IP Packet Format Sent by the I210-CS/CL

L2/L3/L4 Header (updated)	Data (first MSS)	FCS	...	L2/L3/L4 Header (updated)	Data (Next MSS)	FCS	...

Frame formats supported by the I210-CS/CL 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.

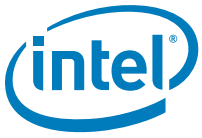
VLAN tag insertion might be handled by hardware

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. The I210-CS/CL has the ability to segment UDP traffic (in addition to TCP traffic), however, because UDP packets are generally fragmented at the IP layer, the I210-CS/CL's “TCP Segmentation” feature is not normally useful to handle UDP traffic.

6.2.4.5 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 6.2.2). 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 6.2.4.6):

Table 6-41. TCP Partial Pseudo-Header Sum for IPv4

IP Source Address		
IP Destination Address		
Zero	Layer 4 Protocol ID	Zero

Table 6-42. 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.

6.2.4.6 Transmit Checksum Offloading with TCP/UDP Segmentation

The I210-CS/CL supports checksum off-loading as a component of the TCP Segmentation off-load feature and as a standalone capability. Section 6.2.5 describes the interface for controlling the checksum off-loading feature. This section describes the feature as it relates to TCP Segmentation.

The I210-CS/CL supports IP and TCP header options in the checksum computation for packets that are derived from the TCP Segmentation feature.

Note: The I210-CS/CL 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. The I210-CS/CL 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 6.2.5 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 6-43. 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 6-44lists the conditions of when checksum off loading can/should be calculated.

Table 6-44. Conditions for Checksum Offloading

Packet Type	IPv4	TCP/UDP	Reason
Non TSO	Yes	No	IP Raw packet (non TCP/UDP protocol)

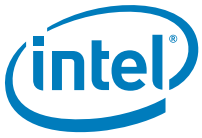


Table 6-44. Conditions for Checksum Offloading

	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

6.2.4.7 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 the I210-CS/CL.

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.

6.2.4.7.1 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

6.2.4.7.2 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

6.2.4.7.3 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)

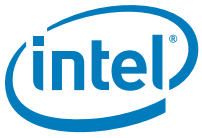
IPv4 Header

- 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

6.2.4.8 Data Flow

The flow used by the I210-CS/CL 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.

6.2.5 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 the I210-CS/CL'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 6.2.2.4.

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 6.2.2, 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.

6.2.5.1 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.

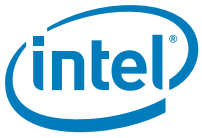
Note: The *MACLEN* + *IPLEN* value needs to be less than the total DMA length for a packet. If this is not the case, the results are unpredictable.

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 6.2.2.2, 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.



6.2.5.2 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.

6.2.5.3 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.

6.2.5.4 Checksum Supported Per Packet Types

Table 6-45 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 6-45. 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[1] Either IP or TCP/SCTP[1]	 Either IP or TCP/SCTP [1] Either IP or TCP/SCTP[1]
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.

6.2.6 Multiple Transmit Queues

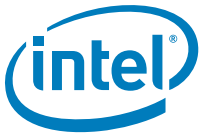
The number of transmit queues is 4, to support Qav functionality or to support load balancing between CPUs.

If there are more CPUs cores than queues, then one queue might be used to service more than one CPU.

For transmission process, each thread might place a queue in the host memory of the CPU it is tied to.

The I210-CS/CL supports assigning either high or low priority to each transmit queue. High priority is assigned to by setting the *TXDCTL[n].priority* bit to 1b. When high priority is assigned to a specific transmit queue, the I210-CS/CL always prioritizes transmit data fetch DMA accesses, before servicing transmit data fetch of lower priority transmit queues. The Audio/Video Bridging (AVB) mechanism described in Section 6.2.7 provides another arbitration mechanisms between the queues.

Note: Throughput of low priority transmit queues can be significantly impacted if high priority queues utilize the DMA resources fully.



6.2.7 Handling Time Sensitive Streams (802.1Qav)

6.2.7.1 Overview

The 802.1Qav is part of the AVB specifications that include Timing and Synchronization for time specific applications (802.1AS), Stream Reservation (SR) protocol to guarantee the resources needed for Audio/Video (AV) streams (802.1Qat), Forwarding and queuing enhancements for time sensitive streams (802.1Qav).

802.1Qav provides a way to guarantee bounded latency and latency variation for time sensitive traffic as AV. It specifies the priority usage and controlled bandwidth draining algorithms.

6.2.7.2 I210-CS/CL Transmit Modes

The I210-CS/CL supports two transmit modes, legacy and Qav. The transmit mode is configured in TQAVCTRL.QavMode register and must be set during the SW initialization cycle. The I210-CS/CL transmit mode cannot change during dynamic operation.

	Legacy	Qav
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	Combination of time based and credit shaper for the SR queues and strict priority for the BE queues

The I210-CS/CL Legacy transmit is defined in Section 6.2. The rest of this sub chapter defines the I210-CS/CL transmit functionality when configured to operate at Qav mode.

Note: When configured to Qav mode enabling the transmit short packet padding feature and sending packets shorter than 64 bytes has some impact to the correctness of the credit shaper arbitration as the arbitration is done based on the non padded transmit packet length.

6.2.7.3 Transmit Architecture in Qav Mode

To enable the proper priority and bandwidth allocation to the time sensitive streams the I210-CS/CL transmit architecture includes up to four transmit packet buffers, the transmit traffic is distributed between the packet buffers based on the packet priority. The I210-CS/CL software device driver directs the outgoing packets to the relevant priority packet buffers by submitting them per priority to the proper descriptor queues.

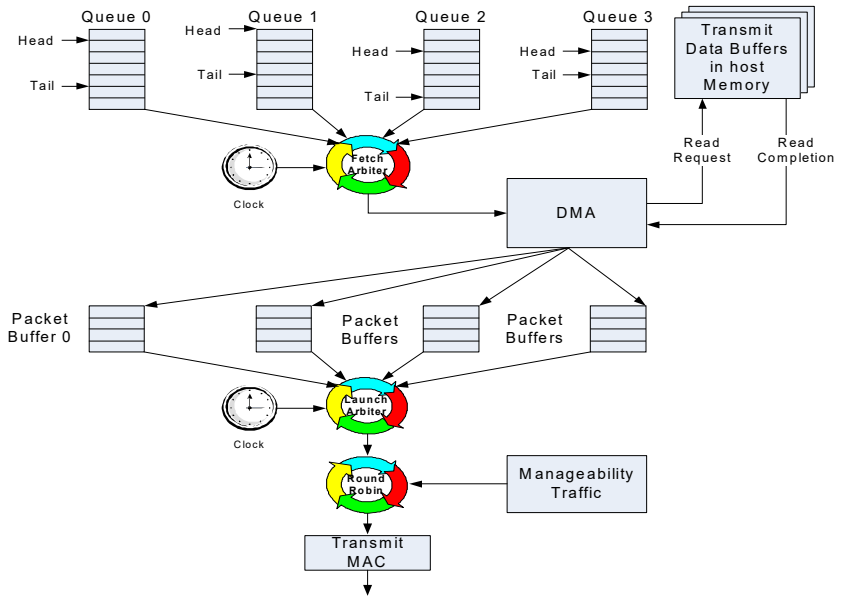


Figure 6-10. Transmit Architecture Qav Mode

6.2.7.4 Mapping User Priorities to Queues

While in Qav mode each active queue in the I210-CS/CL represents a single or multiple User Priorities, The queue priority is pre-assigned such that queue0 has the highest priority and priorities decrease through queue1, queue2 and queue3 has the lowest priority.

Software is responsible to map the proper Ethernet traffic using the appropriate stream priority to its hardware destination queue according to the relevant priority it is assigned.

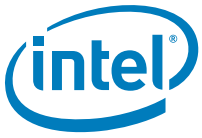
Traffic priority can be assigned according to the VLAN priority field in the VLAN tag.

6.2.7.5 Transmission Selection

Transmission selection is the process of selecting the next packet to transmit, in the I210-CS/CL Qav modes transmission selection includes three levels of arbitration - descriptor fetch, data fetch and data transmission.

Descriptor fetch - the transmit descriptor fetch mechanism while in Qav modes is the same as in legacy mode, the complete description of descriptor fetch is described in Section 6.2.2.5.

Data fetch - the data fetch mechanism while in Qav modes are based on two elements time based and queue arbitration based on round robin or most empty goes first controlled byTQAVCTRL.DataFetchARB. A queue is eligible for arbitrations only if it has descriptors pointing to at least a single packet in host memory. For SR queues with the time based element enabled a queue is only eligible for arbitration if the fetch time of the up coming packet has been reached. See Section 6.2.7.5.3 for more details on how to determine if the fetch time has been reached.



When configured to most empty the queue that wins the arbitration is the queue that is targeted to the most empty packet buffer, for the cases where some packet buffers have the same amount of data (startup for example) arbitration between these queues are done according to the queue priority (higher priority goes first).

Data transmission - transmission arbitration flow is described as follows and shown in Figure 6-11.

- In non Qav mode (TQAVCTRL.TransmitMode = 0b) Arbitration starts from the highest priority queue (index 0) and up to the lowest priority queue (index 3).
- In Qav mode (TQAVCTRL.TransmitMode = 1b) Arbitration starts from the highest priority queue (index 0) and up to the lowest priority queue (index 3) with the following additions:
  - Strict reservation queues (indicated as SR queues or Qav queues) are subjected to credit based shaper criteria (if enabled by the TQAVCTRL.DataTranARB parameter). See credit calculation in Section 6.2.7.5.2.
  - Transmission time base criteria (if enabled by the TQAVCTRL.DataTranTIM parameter). See launch time calculation in Section 6.2.7.5.3.
- Any arbitration step is made for a single packet from the selected queue:
  - If a packet is transmitted, hardware then looks for the next packet to be transmitted from the highest priority queue (index 0).
  - If a packet was not transmitted, hardware then looks for the next packet to be transmitted from the next queue inline.

Note: When in Qav mode, queue0 must be configured as an SR queue, queue1 can be configured as an SR queue or priority queue and queue2, queue3 is configured by default as a priority queue with no ability to be configured as SR queues.

Note: the launch time of a packet is specified in the context descriptor. Every SR packet should be defined using a single context descriptor provided before the packet advanced data descriptors (legacy descriptors are not supported in Qav mode).

6.2.7.5.1 Data Transmission Arbitration Algorithm

Definition and description of parameters

Priority (QN - Queue number), the I210-CS/CL in Qav mode implements up to 4 priorities defined by the actual queue used, see Section 6.2.7.4 for the way priorities are mapped to queues

Credits - Regulates the bandwidth allocation to user priorities, credits represent a single byte. The transmission of a queue in SR mode is defined by the amount of credits assigned to that queue

QueueFrames - Each queue has an indication whether there are queued frames for that queue

Launch Time - Defines the time to launch the packet for time based arbitration

Mode - While in Qav mode each queue is configured to be either in priority mode or SR mode

Table 6-46 lists the arbitration modes as controlled by the TQAVCTRL register followed by a flow diagram that illustrates the arbitration scheme for Qav mode.

Table 6-46. Transmission Arbitration

TransmitMode	DataTranARB	DataTranTIM	Functionality
0 (Legacy)	X	X	Single packet buffer for all 4 queues. Packets are transmitted at the same order they are fetched from host memory.
1 (Qav)	0 (Strict Priority)	0	Strict priority queuing without any scheduling.
1 (Qav)	0 (Strict Priority)	1 (Launch Time)	Strict priority queuing while SR queues are subjected to launch time policy.
1 (Qav)	1 (Credit Shaper)	0	Strict priority queuing while SR queues are subjected to Credit Shaper policy.
1 (Qav)	1 (Credit Shaper)	1 (Launch Time)	Strict priority queuing while SR queues are subjected to Credit Shaper policy plus launch time.

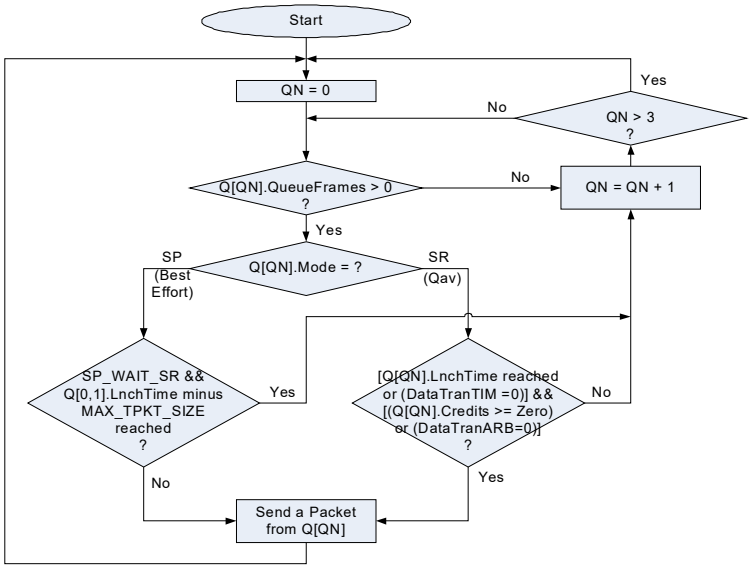


Figure 6-11. Data Transmission Arbitration Operation for TransmitMode = Qav

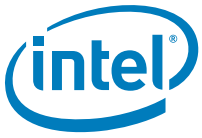
6.2.7.5.2 Data Transmission Credit Calculation

Definition and description of parameters

IdleSlope - The Idle slop of an SR queue defines the amount of credit accumulation (per byte) while the queue is blocked from transmission

SendSlope - The Send slop of an SR queue defines the amount of credit consumption (per byte) and is calculated by the formula SendSlope = IdleSlope - LinkRate

HiCredit - The HiCredit of an SR queue defines the maximum credit accumulation for that queue



Credit calculation pseudo code:

```
For each Q[i]
    If Transmit
        Credit = Credit – SendSlope
    If !Transmit
        If Q[i]QuFrm = 0
            Credit = 0
        Else If Credit + IdleSlope > HiCredit
            Credit = HiCredit
        Else Credit = Credit + IdleSlope
```

Note: During QavMode TIPG register value should not be modified from its default value of total 12 bytes IPG between packets.

6.2.7.5.3 Launch Time/Fetch Time Decision

Launch time and Fetch time criteria is defined to be "pass" if either:

- The Launch time/Fetch time match exactly the relevant portion of SYSTIML value
  - It is compared against SYSTIML[29:5], and provides transmission granularity of 0.032 µs
- The time passed from (post) Launch time/Fetch time in the SYSTIML is within the allowed Launch time/Fetch time criteria
  - The Allowed Fetch time and Allowed Launch time should be calculated such that it is allowed to Fetch/Transmit a packet if the current time is within Fetch/Launch time + 0.5 second.
  - Note that the SYSTIML register max value is 999,999,999 dec (0x3B9AC9FF) and it wraps to 0 when reaching this value (representing a full second).

6.2.7.6 Qav Latency

The latency between transmission scheduling (either credit base or launch time) and the time the packet is transmitted to the network is listed in Table 6-61.

Table 6-47. Packet Scheduling to its Transmission Latency

Link Speed	Latency between launch time and packet in the MAC (the time on which packet 1588 time stamping is taken). For a complete delay from between launch time and SFD on the PHY MDI pins, please add the latency defined in Table 6-61 (delay between packet timestamp and SFD on the MDI pins)		
	Min	Max	Comments
Transmit at 10 Mb/s	Not measured		Assume no interfering transmission (see also SP_WAIT_SR setting). The min/max values represent possible jitter. In case of no concurrent receive, the jitter is reduced by 16 ns.
Transmit at 100 Mb/s	1.184 µs	1.360 µs	
Transmit at 1 Gb/s	288 ns	304 ns	

6.2.7.7 Qav Configuration

Context Descriptor Configuration

LaunchTime (25 bits (56:32) based on SYSTIM)

Global Qav configuration:

TXPBSIZE.Tx0pbsize/Tx1pbsize/Tx2pbsize/Tx3pbsize (Tx packet buffer size assignment)

- Recommended configuration is 8KB for PB0, PB1 and 4KB for PB2, PB3

RXPBSIZE.Rxpbsize (Rx packet buffer size assignment)

- To comply with the Tx recommendation above need to set to 0x20 (32 KB). Refer to the setting rule defined in Section 4.5.9.

DTXMXPKTSZ.MAX_TPKT_SIZE (DMA Tx maximum packet size)

TQAVCTRL.TransmitMode (Transmit mode configuration: legacy, Qav)

TQAVCTRL.DataFetchArb (Data fetch arbitration configuration: Round Robin, Most empty)

- TXDCTL.Priority can be use to prioritize SR queues over SP queues

TQAVCTRL.DataTranArb (Data Transmit arbitration configuration: Strict Priority, Credit Shaper Algorithm)

TQAVCTRL.DataTrantim (Data Transmit Time Valid configuration - controls time based transmission)

TQAVCTLRL.FetchTimDelta (Fetch Time Delta configuration - the time to reduce from Launch time to make a SR queue packet eligible for fetch)

Per Queue (0/1) Qav configuration:

TQAVCC.QueueMode (Queue mode configuration: SR, SP)

TQAVCC.IdleSlope (Idle slope configuration credits)

IdleSlope Configuration is calculated using the following equation:

100Mbps: BW*0x7735*0.2

1000Mbps: BW*0x7735*2

BW is the percentage BW out of full line rate.

TQAVHC.HiCredit (Maximum number of credits that can be accumulated per queue)

LinkRate (Not configured and always defaults to 0x7735 credits)

ZeroCredit (not configured and always defaults to 0x80000000)

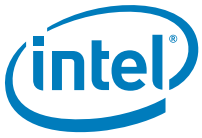
SW calculations of credit limits for proper setting:

SR0 (Queue0):

- HiCredit = 0x80000000 + MAX_TPKT_SIZE*Idle_BW[Queue0]*0x7735

SR1 (Queue1):

- HiCredit = Max{(0x80000000 + 2*MAX_TPKT_SIZE*Idle_BW[Queue1]*0x7735),(0x80000000 + ((Idle_BW[Queue0]/Send_BW[Queue0])*MaxFameSize + MAX_TPKT_SIZE)*Idle_BW[Queue1]*0x7735)}



6.3 Interrupts

6.3.1 Interrupt Modes

The I210-CS/CL 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.

6.3.1.1 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 the I210-CS/CL interrupt causes into an interrupt vector that is conveyed by the I210-CS/CL 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.

6.3.2 Mapping of Interrupt Causes

There are 10 extended interrupt causes that exist in the I210-CS/CL:

1. 8 traffic causes — 4 Tx, 4 Rx.
2. TCP timer
3. Other causes — Summarizes legacy interrupts into one extended cause.

The way the I210-CS/CL 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.

6.3.2.1 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 6-12 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.

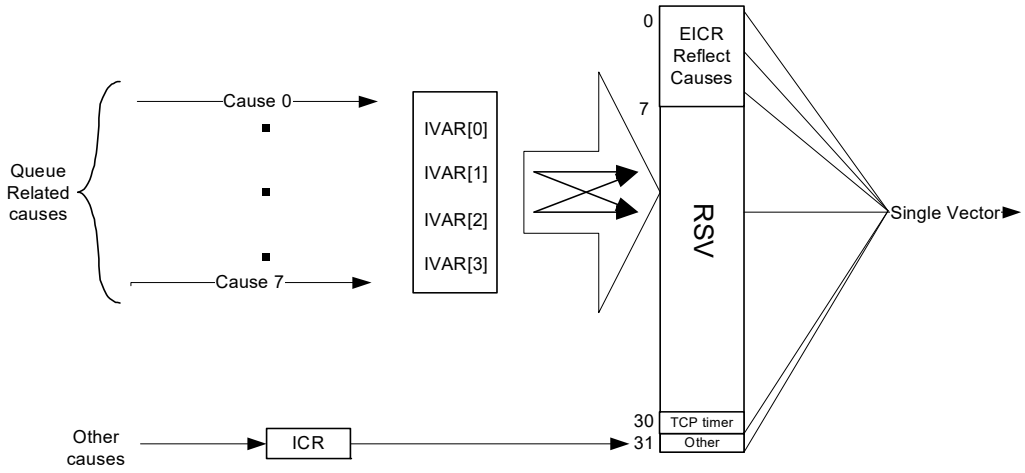


Figure 6-12. Cause Mapping in Legacy Mode

Table 6-48 lists the different interrupt causes into the IVAR registers.

Table 6-48. 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.

6.3.2.2 MSI-X Mode

In MSI-X mode the I210-CS/CL 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 6-13 shows the allocation process.

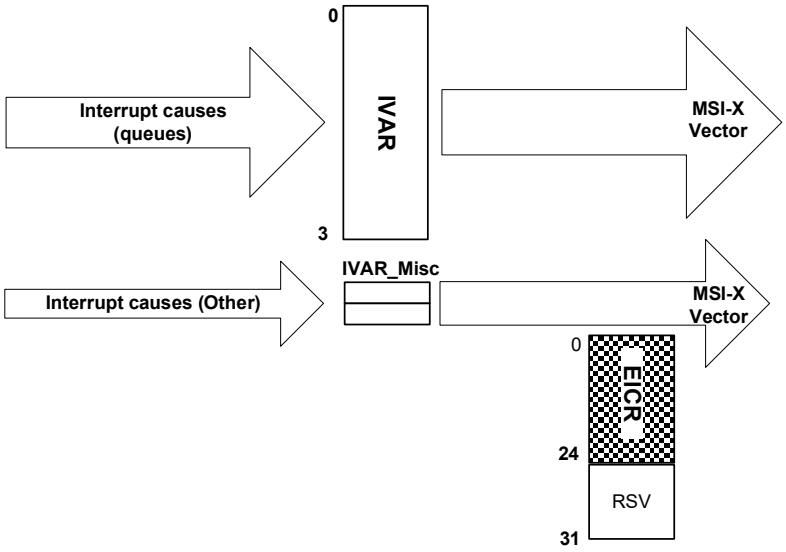


Figure 6-13. Cause Mapping in MSI-X Mode

Table 6-49 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 6-49. 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.

6.3.3 Legacy Interrupt Registers

The interrupt logic consists of the registers listed in Table 6-50 and Table 6-51, plus the registers associated with MSI/MSI-X signaling. Table 6-50 lists the use of the registers in legacy mode and Table 6-50 lists the use of the registers when using the extended interrupts functionality

Table 6-50. 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 6-51. 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.

6.3.3.1 Interrupt Cause Register (ICR)

6.3.3.1.1 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.

6.3.3.1.2 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.

6.3.3.2 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.

6.3.3.3 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.

6.3.3.4 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).

6.3.3.5 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.

6.3.3.6 Extended Interrupt Cause Registers (EICR)

6.3.3.6.1 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.

6.3.3.6.2 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.

6.3.3.7 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.

6.3.3.8 Extended Interrupt Mask Set and Read Register (EIMS) & Extended Interrupt Mask Clear Register (EIMC)

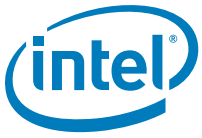
Interrupts appear on PCIe only if the interrupt cause bit is a one and the corresponding interrupt mask bit is a one. Software blocks assertion of an interrupt by clearing the corresponding bit in the mask register. The cause bit stores the interrupt event regardless of the state of the mask bit. Different 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.

6.3.3.9 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 6.3.4 for additional information on the interrupt cause reset process.



6.3.3.10 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.

6.3.3.11 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 6-52 lists the recommended setting of these bits in the different modes:

Table 6-52. Settings for Different Interrupt Modes

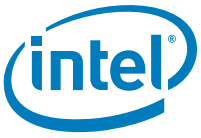
Field	Bit(s)	Initial Value	Description	INT-x/ MSI + Legacy	INT-x/ MSI + Extend	MSI-X Multi Vector	MSI-X Single Vector
NSICR	0	0b	Non Selective Interrupt clear on read: When set, every read of the ICR register clears the ICR register. When this bit is cleared, an ICR register read causes the ICR register 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, the I210-CS/CL 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.

6.3.4 Clearing Interrupt Causes

The I210-CS/CL 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.



6.3.4.1 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.

6.3.4.2 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.

6.3.4.3 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).

6.3.5 Interrupt Moderation

An interrupt is generated upon receiving of incoming packets, as throttled by the EITR registers (see Section 7.8.14). There is an *EITR* register per MSI-X vector.

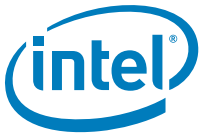
In MSI-X mode, each active bit in EICR can trigger the interrupt vector it is allocated to. Following the allocation, the EITR corresponding to the MSI-X vector is tied to one or more bits in EICR.

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 the I210-CS/CL 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:

$$\text{inter-interrupt interval} = (1 * 10^{-6} \text{ sec x interrupt/sec})^{-1}$$

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.

The I210-CS/CL 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 6-14.

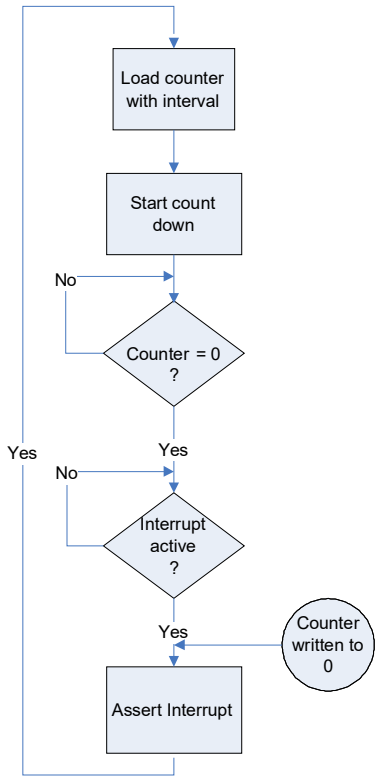


Figure 6-14. Interrupt Throttle Flow Diagram

EITR is designed to guarantee the total number of interrupts per second so for cases where the I210-CS/CL 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.

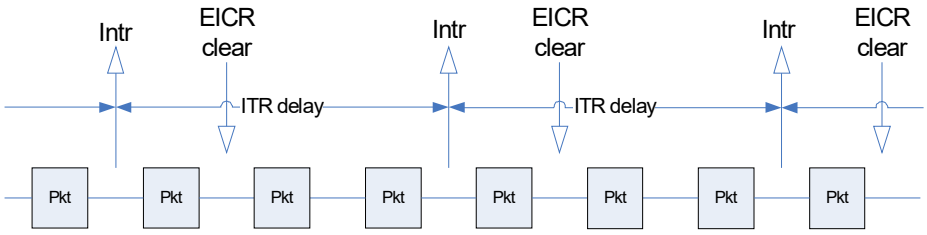


Figure 6-15. Case A: Heavy Load, Interrupts Moderated

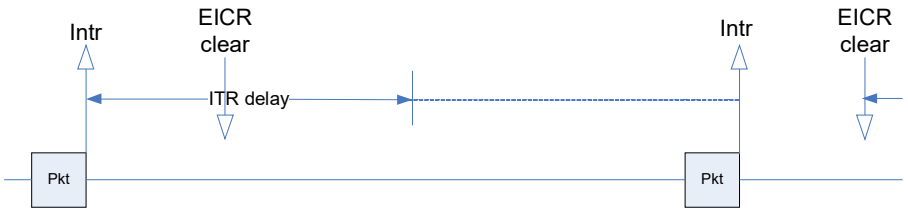


Figure 6-16. Light load, Interrupts Immediately on Packet Receive

6.3.6 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 2-tuple 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 6.1.1. 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 2-tuple filters. The content of each filter is described in Section 6.1.2.4. 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 6.1.2.5. 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 2-tuple or Flex filters.
- The SYN packets filter defined in Section 6.1.2.6 and the ethernet type filters defined in section Section 6.1.2.3 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.

6.3.6.1 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, the I210-CS/CL 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

6.3.7 TCP Timer Interrupt

6.3.7.1 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.

6.3.7.2 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 7.8.3). The counter re-starts counting from its initial value if the *TCPTIMER.Loop* field is set.

6.3.8 Setting Interrupt Registers

In each mode, the registers controlling the interrupts should be set in a different way to assure the right behavior.

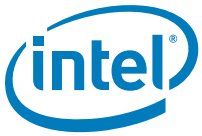


Table 6-53. Registers Settings for Different Interrupt Modes

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 6-52 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.

6.4 802.1Q VLAN Support

The I210-CS/CL 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.

6.4.1 802.1Q VLAN Packet Format

The following diagram compares an untagged 802.3 Ethernet packet with an 802.1Q VLAN tagged packet:

Table 6-54. 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.

6.4.2 802.1Q 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 7.10.1.
- VLAN Identifier (VID)

The bit ordering is as follows:

Table 6-55. TCI Bit Ordering

Octet 1								Octet 2							
UP			CFI	VID											

6.4.3 Transmitting and Receiving 802.1Q Packets

6.4.3.1 Adding 802.1Q Tags on Transmits

Software might command the I210-CS/CL 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 the I210-CS/CL 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 6-8) of the descriptor. Refer to Table 6-26, 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 6.2.2.2.1) of the advanced context descriptor. The *IDX* field of the advanced Tx descriptor should be set to the adequate context.

6.4.3.2 Stripping 802.1Q Tags on Receives

Software might instruct the I210-CS/CL 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 the I210-CS/CL strips the 4 byte VLAN tag from the packet, and stores the TCI in the VLAN *Tag* field (see Figure 6-4 and See "Receive UDP Fragmentation Checksum) of the receive descriptor.

The I210-CS/CL 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.

VLAN stripping can be enabled using two different modes:

1. By setting the *DVMOLR.STRVLAN* for the relevant queue.

2. By setting the *CTRL.VME* bit.

6.4.4 802.1Q 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.

The I210-CS/CL provides exact VLAN filtering for VLAN tags for host traffic.

6.4.4.1 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 7.10.18.

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.

6.4.5 Double VLAN Support

The I210-CS/CL 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* Flash word.

The type of the VLAN tag used for the additional VLAN is defined in the *VET.VET_EXT* field.

6.4.5.1 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 6.2.2 for details). The I210-CS/CL 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 the I210-CS/CL 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 using the switch based VLAN insertion mechanism or from the descriptor (see Section 6.4.3.1), 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.

6.4.5.2 Receive Behavior With External VLAN

When the I210-CS/CL is working in this mode, it assumes that all packets received have at least one VLAN.

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 6-56 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 the I210-CS/CL 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 6.1.1.

Table 6-56. 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.

6.5 Configurable LED Outputs

The I210-CS/CL implements 3 output drivers intended for driving external LED circuits. Each of the 3 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 non-blinking (steady-state) indication.

The configuration for LED outputs is specified via the *LEDCTL* register. Furthermore, the hardware-default configuration for all the LED outputs, can be specified via Flash fields, thereby supporting LED displays configurable to a particular OEM preference.

Each of the 3 LED's might be configured to use one of a variety of sources for output indication. The MODE bits control the LED source as described in Table 6-57.

The IVRT bits allow the LED source to be inverted before being output or observed by the blink-control logic. LED outputs are assumed to normally be connected to the negative side (cathode) of an external LED.

The BLINK bits control whether the LED should be blinked (on for 200ms, then off for 200ms) while the LED source is asserted. The blink control might be especially useful for ensuring that certain events, such as ACTIVITY indication, cause LED transitions, which are sufficiently visible by a human eye.

Note: When LED Blink mode is enabled the appropriate LED Invert bit should be set to 0b.

The LINK/ACTIVITY source functions slightly different from the others when BLINK is enabled. The LED is off if there is no LINK, on if there is LINK and no ACTIVITY, and blinking if there is LINK and ACTIVITY.

The dynamic LED modes (FILTER_ACTIVITY, LINK/ACTIVITY, COLLISION, ACTIVITY, PAUSED) should be used with LED Blink mode enabled.

6.5.1 MODE Encoding for LED Outputs

Table 6-57 lists the MODE encoding for LED outputs used to select the desired LED signal source for each LED output.

Table 6-57. Mode Encoding for LED Outputs

Mode	Selected Mode	Source Indication
0000b	LINK_10/1000	Asserted when either 10 or 1000 Mb/s link is established and maintained.
0001b	LINK_100/1000	Asserted when either 100 or 1000 Mb/s link is established and maintained.
0010b	LINK_UP	Asserted when any speed link is established and maintained.
0011b	FILTER_ACTIVITY	Asserted when link is established and packets are being transmitted or received that passed MAC filtering.
0100b	LINK/ACTIVITY	Asserted when link is established and when there is no transmit or receive activity. When BLINK, indicates LINK and activity (eIther receive or transmit)
0101b	LINK_10	Asserted when a 10 Mb/s link is established and maintained.
0110b	LINK_100	Asserted when a 100 Mb/s link is established and maintained.
0111b	LINK_1000	Asserted when a 1000 Mb/s link is established and maintained.
1000b	SDP_MODE	LED activation is a reflection of the SDP signal. SDP0, SDP1, SDP2 are reflected to LED0, LED1, LED2 respectively.

Table 6-57. Mode Encoding for LED Outputs (Continued)

Mode	Selected Mode	Source Indication
1001b	FULL_DUPLEX	Asserted when the link is configured for full duplex operation (de-asserted in half-duplex).
1010b	COLLISION	Asserted when a collision is observed.
1011b	ACTIVITY	Asserted when link is established and packets are being transmitted or received.
1100b	LINK_10/100	Asserted when either 10 or 100 Mb/s link is established and maintained.
1101b	PAUSED	Asserted when the I210-CS/CL's transmitter is flow controlled.
1110b	LED_ON	Always high (Asserted)
1111b	LED_OFF	Always low (De-asserted)

6.6 Memory Error Correction and Detection

The I210-CS/CL 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.

The I210-CS/CL 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 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.
   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.
   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 *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.

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* Flash word (See Section 6.2.2) or by clearing the *PCIEERRCTL.GPAR_EN* bit (See Section 7.22.4).

6.6.1 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.

To recover from a parity error event software should initiate the following actions depending on the region in which the parity error occurred.

6.6.1.1 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.RST* bit.
2. wait at least 3 milliseconds after setting *CTRL.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.

6.6.1.2 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.RST* bit as specified in Section 4.3.1 and re-initializing the port.

6.6.1.3 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 7.22.11 (*LANPERRSTS* register) according to the ram that failed.

6.7 CPU Affinity Features

6.7.1 Direct Cache Access (DCA)

6.7.1.1 DCA Description

Direct Cache Access (DCA) is a method to improve network I/O performance by placing some posted inbound writes indirectly within CPU cache. DCA requires that memory writes go to host memory and then the processor prefetch the cache lines specified by the memory write. Through research and experiments, DCA has been shown to reduce CPU Cache miss rates significantly.

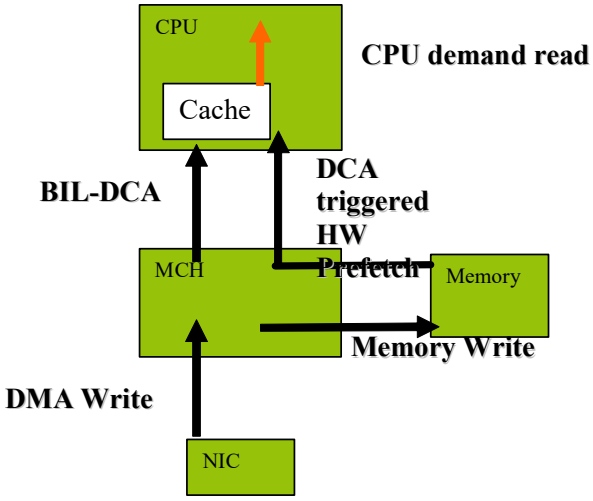


Figure 6-17. Diagram of DCA Implementation on FSB System

As shown in Figure 6-17, DCA provides a mechanism where the posted write data from an I/O device, such as an Ethernet NIC, can be placed into CPU cache with a hardware pre-fetch. This mechanism is initialized upon a power good reset. A software device driver for the I/O device configures the I/O device for DCA and sets up the appropriate DCA target ID for the device to send data. The device then encapsulates that information in PCIe TLP headers, in the *TAG* field, to trigger a hardware pre-fetch by the MCH /IOH to the CPU cache.

DCA implementation is controlled by separated registers (*RXCTL* and *TXCTL*) for each receive and transmit queue. In addition, a *DCA Enable* bit can be found in the *DCA_CTRL* register, and a *DCA_ID* register, in order to make visible the function, device, and bus numbers to the driver.

The *RXCTL* and *TXCTL* registers can be written by software on the fly and can be changed at any time. When software changes the register contents, hardware applies changes only after all the previous packets in progress for DCA have been completed.

However, in order to implement DCA, the I210-CS/CL has to be aware of the Crystal Beach version used. Software driver must initialize the I210-CS/CL to be aware of the Crystal Beach version. A register named *DCA_CTRL* is used in order to properly define the system configuration.

There are 2 modes for DCA implementation:

1. Legacy DCA: The DCA target ID is derived from CPU ID.
2. DCA: The DCA target ID is derived from APIC ID.

The software driver selects one of these modes through the DCA_mode register.

The details of both modes are described in the following sections.

6.7.1.2 Details of Implementation

6.7.1.2.1 PCIe Message Format for DCA

Figure 6-18 shows the format of the PCIe message for DCA.

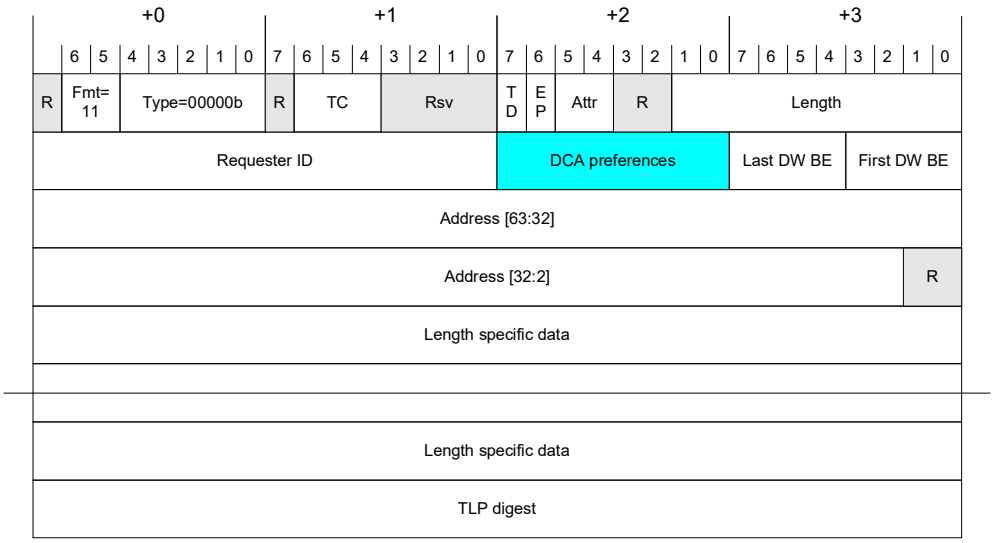


Figure 6-18. PCIe Message Format for DCA

The DCA preferences field has the following formats.

Table 6-58. Legacy DCA Systems

Bits	Name	Description
0	DCA indication	0b: DCA disabled 1b: DCA enabled
3:1	DCA Target ID	The DCA Target ID specifies the target cache for the data.
7:4	Reserved	Reserved

Table 6-59. DCA Systems

Bits	Name	Description
7:0	DCA target ID	0000.0000b: DCA is disabled Other: Target Core ID derived from APIC ID.

6.7.2 TLP Process Hints (TPH)

The I210-CS/CL supports the TPH capability defined in the PCI Express specification (See Section 8.5). It does not support Extended TPH requests.

On the PCIe link existence of a TLP Process Hint (TPH) is indicated by setting the *TH* bit in the TLP header. Using the PCIe TLP Steering Tag (ST) and Processing Hints (PH) fields, the I210-CS/CL can provide hints to the root complex about the destination (socket ID) and about data access patterns (locality in Cache), when executing DMA memory writes or read operations. Supply of TLP Processing Hints facilitates optimized processing of transactions that target Memory Space.

The I210-CS/CL supports a steering table with 8 entries in the PCIe TPH capability structure (See Section 8.5.3.4). The PCIe Steering table can be used by Software to provide Steering Tag information to the Device via the *TXCTL.CPUID* and *RXCTL.CPUID* fields.

To enable TPH usage:

1. For a given function, the *TPH Requester Enable* bit in the PCIe configuration *TPH Requester Control Register* should be set.
2. Appropriate TPH Enable bits in *RXCTL* or *TXCTL* registers should be set.
3. Processing hints should be programmed in the *DCA_CTRL.Desc_PH* and *DCA_CTRL.Data_PH* Processing hints (PH) fields.
4. Steering information should be programed in the CPUID fields in the *RXCTL* and *TXCTL* registers.

The Processing Hints (PH) and Steering Tags (ST) are set according to the characteristics of the traffic as described in Table 6-60.

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

Note: Per queue, the DCA and TPH features are exclusive. Software can enable either the DCA feature or the TPH feature for a given queue.

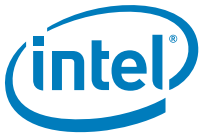
6.7.2.1 Steering Tag and Processing Hint Programming

Table 6-60 lists how the Steering tag (socket ID) and Processing hints are generated and how TPH operation is enabled for different types of DMA traffic.

Table 6-60. Steering Tag and Processing Hint Programming

Traffic Type	ST Programming	PH Value	Enable
Transmit descriptor write back or head write back	TXCTL.CPUID[1]	*DCA_CTRL.Desc_PH*[2]	*Tx Descriptor Writeback TPH EN* field in *TXCTL*.
Receive data buffers write	RXCTL.CPUID[1]	*DCA_CTRL.Data_PH*[3]	*RX Header TPH EN or Rx Payload TPH EN* fields in *RXCTL.*
Receive descriptor writeback	RXCTL.CPUID[1]	*DCA_CTRL.Desc_PH*[2]	*RX Descriptor Writeback TPH EN* field in *RXCTL.*
Transmit descriptor fetch	TXCTL.CPUID[4]	*DCA_CTRL.Desc_PH*[2]	*Tx Descriptor Fetch TPH EN* field in *TXCTL*.
Receive descriptor fetch	RXCTL.CPUID[2]	*DCA_CTRL.Desc_PH*[2]	*Rx Descriptor fetch TPH EN* field in *RXCTL*.
Transmit packet read	TXCTL.CPUID[2]	*DCA_CTRL.Data_PH*[3]	*Tx Packet TPH EN field in TXCTL*.

1. the driver should always set bits [7:3] to zero and place Socket ID in bits [2:0].
2. Default is 00b (Bidirectional data structure).
3. Default is 10b (Target).
4. the hints are always zero.



6.8 Time SYNC (IEEE1588 and IEEE 802.1AS)

6.8.1 Overview

IEEE 1588 addresses the clock synchronization requirements of measurement and control systems. The protocol supports system-wide synchronization accuracy in the sub-microsecond range with minimal network and local clock computing resources. The protocol is spatially localized and allows simple systems to be installed and operate.

The IEEE802.1AS standard specifies the protocol used to ensure that synchronization requirements are met for time sensitive applications, such as audio and video, across bridged and Virtual Bridged Local Area Networks (VBLAN) consisting of LAN media where the transmission delays are almost fixed and symmetrical. For example, IEEE 802.3 full duplex links. This includes the maintenance of synchronized time during normal operation and following addition, removal, or failure of network components and network re-configuration. It specifies the use of IEEE 1588 specifications where applicable.

Activation of the I210-CS/CL Time Sync mechanism is possible in full duplex mode only. No limitations on wire speed exist, although wire speed might affect the accuracy. Time Sync protocol is tolerant of dropping packets as well as missing timestamps.

6.8.2 Flow and Hardware/Software Responsibilities

The operation of a PTP (Precision Time Protocol) enabled network is divided into two stages, initialization and time synchronization. These stages are described in the sections that follow emphasizing hardware and software roles.

6.8.2.1 Initialization Phase

At the initialization stage the software on every master enabled node starts by sending Sync packets that include its clock parameters. Upon reception of a Sync packet a node, the software on any potential master, compares the received clock parameters to its own parameters. If the received clock parameters of a peer are better, the software transits to Slave state and stops sending Sync packets. When in slave state, the software selects a particular master. It compares continuously the received Sync packet to its selected master. If the received Sync packets belong to a different master with better clock parameters, the software on the slave switches to the new master. Eventually only one master (with the best clock parameters) remains active while all other nodes act as slaves listening to that master. Every node has a defined Sync packet time-out interval. If no Sync packet is received from its chosen master clock source during the interval the software on the master enabled nodes transit back to master state at initialization phase. Note that there are more than one option for the above flow. For example, one node could be set statically as the master while all other notes are set as slaves listening to that master.

6.8.2.2 Time Synchronization Phase

There are two phases to the synchronization flow: At the beginning, the slave calibrates its clock to the master and then it performs the complete synchronization.

6.8.2.2.1 2-step Clocks Calibration Procedure

The master send SYNC packets periodically (in the order of 10 packets per second). These packets are followed by Follow_UP packets that indicate the transmission time. The slave captures the reception time of these SYNC packets. Together with the Follow_UP packets the slave holds the SYNC packet transmission time at the master and its reception time at the slave. The slave calculates the time gap between consecutive SYNC packets defined by the master clock. It then calibrates itself to get the same time gap defined by its own clock. During this phase the slave also sets its time to be as close as possible to the master time (as accurate as the transmission delay and software latencies).

In order to minimize sampling inaccuracy, both master and slave sample the packets transmission and reception time at a location in the hardware that has as much as possible deterministic delay from the PHY interface.

Packet processing in the master and the Slave

- The master software indicates the SYNC packet to the hardware by setting the 2STEP_1588 flag in the Advanced Transmit Data Descriptor. Setting this flag, the hardware samples its transmission time by the TXSTMP register. The software reads its value and sends the transmission time in a Follow_Up packet.
- The SYNC packet is received by the slave and its reception time is posted to the "timestamp bytes" in the packet buffer in host memory. The Follow_Up packet is also received and posted to the software processing. The software uses these parameters to calculate the time gap between consecutive packets by its own clock compared to the master clock taking the required corrective action.

6.8.2.2.2 1-step Clocks Calibration Procedure

The I210-CS/CL supports the 1-step procedure. In 1-step procedure, the hardware inserts the transmission time to the sent SYNC packets at the master (as follows). All the rest is the same as the 2-step procedure described above.

- The software indicates the SYNC packets to the hardware by setting the 1STEP_1588 flag in the Advanced Transmit Data Descriptor. Setting this flag, the hardware does the following:
  - Samples the packet transmission time
  - Auto-inserts the packet transmission timestamp at the offset defined by the *1588_Offset* field in the *TSYNCTXCTL* register
  - Insert the Ethernet CRC while including the timestamp in the CRC calculation
  - The UDP checksum is not updated by the inserted timestamp. It means that 1-step is limited for PTP over L2 or PTP over UDP/IPv4 while the UDP checksum is not used (equals to zero).

6.8.2.2.3 2-step Time Synchronization Phase Procedure

The complete synchronization scheme is shown in Figure 6-19. It relies on measured timestamp of Sync packets transmission and reception by the master and the slave. The scheme is based on the following two basic assumptions:

- The clocks at both nodes are almost identical (achieved in the first step)
- Transmission delays between the master to the slave and backward are symmetric

The master's software sends periodically Sync packets to each slave followed by the Follow_Up packet (as explained in the Clocks Calibration (2-step procedure). The responds back by sending Delay_Req packets which are sampled by the slave and the master. The master provides back its parameters which are used by the slave to calibrate its time. Following are the detailed software hardware steps.

Packet processing in the master and the slave:

- The master software sends the SYNC packet and Follow_Up packet as described in the Clocks Calibration (2-step procedure) procedure.
- Processing these packets by the slave is also the same as the Clocks Calibration (2-step procedure) procedure
- The slave software responds back by sending the Delay_Req packet (for those SYNC packets that the slave "wish" to respond). The Delay_Req packet is indicated to the hardware by setting the 2STEP_1588 flag in the Advanced Transmit Data Descriptor. The transmission time is extracted from the TXSTMP register the same as the master processes the transmitted SYNC packets.
- The Master receives the Delay_Req packet and its reception time is posted to the "timestamp bytes" in the packet buffer in host memory.
- The master software sends back the received timestamp to the slave which has all required timestamps.
- The slave adjust its time according to the following equation (or a low-pass version of the equation):

Slave Adjust Time = - [(T2-T1) - (T4-T3)] / 2
While using the following notations:
- **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)

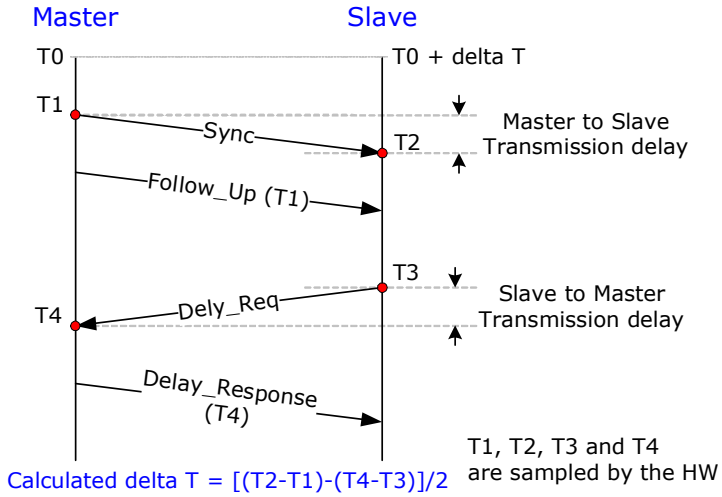


Figure 6-19. Sync Flow and Offset Calculation

6.8.2.2.4 1-step Time Synchronization Phase Procedure

Packet processing in the master and the Slave for 1-step procedure is almost identical to the 2-step procedure as follow:

- The master software sends the SYNC packet while indicating it to the hardware by the 1STEP_1588 flag. Doing so, the hardware inserts the transmission time in the SYNC packet.
- The slave samples the reception time of the SYNC and extract its transmission time at the master.
- From this point the flow is identical to the 2-step procedure.

6.8.2.3 TimeSync Indications in Receive and Transmit Packet Descriptors

Certain indications are transferred between software and hardware regarding PTP packets. These indications are enabled when the *Disable systime* bit in the *TSAUXC* register is cleared. Further more, transmit timestamping is enabled by the *TSYNCTXCTL.EN* flag. Received packets for captured time are identified according to the *TSYNCRXCTL.Type* and *CTRLT* and *MSGT* fields in the *TSYNCRXCFG* register.

2-step SYNC and Delay_Req packet transmission: The software sets the *2STEP_1588* bit in the Advanced Transmit Data Descriptor. The hardware samples the transmission time in the TXSTMP register. The software reads it for every packet and used the transmission time as required.

1-step SYNC packet transmission: On the transmit path the software sets the *1STEP_1588* bit in the Advanced Transmit Data Descriptor. It should also set previously the *1588_Offset* field in the *TSYNCTXCTL* register. The hardware samples the transmission time and inserts it in the transmitted packet at the offset defined by the 1588_Offset field. The software should prepare the space in the transmitted packet by filling it with zero's while the hardware replaces these zero's by the transmission timestamp. The transmission time stamp is an 80-bit field while the 32 LS bits specify the transmission time in nsec units and the upper 48 bits specify the time in second units. Note that the 1588 timer in the I210-CS/CL contains only 32 bits that specify the second units. The additional upper 16 bits are taken from the static SYSTIMTM register which is set by software (expected to be zero at all times). Timestamp transmission on the wire is as follows: The MS byte of the SYSTIMTM register is transmitted first while the LS byte of the nsec units is transmitted last (as shown in Table 6-66).

Packet reception: PTP packet identification is described in Section 6.8.5. L2 packets that are identified by the EtherType are indicated by the "*packet type*" field in the receive descriptor. Those packets that the hardware samples its reception time are also indicated by the *TS* or *TSIP* flags in the advanced receive descriptors. Selecting between *TS* or *TSIP* reporting is controlled by the *Timestamp* flag in the *SRRCTL[n]* register (per receive queue). If the *TS* flag is set, the packet reception time is sampled by the hardware in the *RXSTMPL/H* registers. These registers are locked until the software reads its value. If the *TSIP* flag is set, the packet reception time is posted to the packet buffer in host memory as shown in Section 6.1.6.

6.8.3 Hardware Time Sync Elements

All time sync hardware is initialized as defined in the registers section upon MAC reset. The time sync logic is enabled if the *TSAUXC.Disable systime* flag is cleared.

The 1588 logic includes multiple registers larger than 32 bits which are indicated as xxxL (Low portion - LS) and xxxH (High portion - MS). When software accesses these registers (either read or write) it should access first the xxxL register (LS) and only then the xxxH register (MS). Accessing the xxxH might impact the hardware functionality which should be triggered only after both portions of the register are valid.

6.8.3.1 Capture Timestamp Mechanism

The timestamp logic is located on transmit and receive paths as close as possible to the PHY interface. The timestamp is captured at the beginning of the packet as shown in the Figure 6-20. These rules keep the latency between the captured timestamp and transmission time as deterministic as possible. The 1588 logic is functional at all link speeds; however, the latency parameters characterized at this time is only for 100 Mb/s. The measured latency parameters in a stand-alone setup are listed in Table 6-61. When measured against a commercial link partner using an arithmetic mean and exponential smoothing, a shift of approximately 40 ns is used as listed in Table 6-62.

Table 6-61. Packet Timestamp Sampling Latency at 100 Mb/s (Stand-alone Setup)

Parameter	Min/Max Latency	Comment
Tx timestamp to start of SFD on MDI	984/1024 ns	Min/max values represent a possible variance over reset or link up/down events. The latency is measured with minimal PHY FIFO depth by setting bits 15:14 in the PHY TX FIFO register (MAC Specific Control Register 1 - Page 2, Register 16). Setting the PHY Tx FIFO to other values increase the delay by an 8-bit time for each increment of the FIFO size.
Start of SDF on MDI to Rx timestamp	2148/2228 ns	The min/max numbers represent possible jitter due to synchronization between receive and transmit clock domains.

Table 6-62. Packet Timestamp Sampling Latency at 100 Mb/s (Arithmetic Mean and Exponential Smoothing)

Parameter	Average	Comment
Tx timestamp to start of SFD on MDI	1044 ns	The range (max minus min) values measured for the Tx and Rx latency parameters are similar to the measured parameters in a stand-alone setup.
Start of SDF on MDI to Rx timestamp	2133 ns	
Tx + Rx latency	3177 ns	

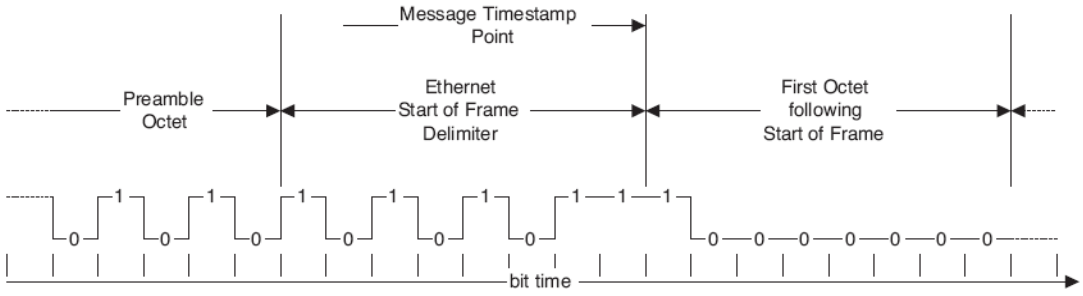


Figure 6-20. Timestamp Point

6.8.3.2 1588 Timer Registers: SYSTIM, TIMADJ and TIMINCA

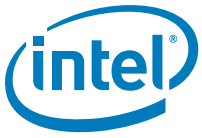
The *SYSTIM* is a 96-bit register is composed of: *SYSTIMR*, *SYSTIML* and *SYSTIMH* registers: The *SYSTIMR* register holds the sub ns fraction, the *SYSTIML* register holds the ns fraction and the *SYSTIMH* register holds the second fraction of the time (note that the upper two bits of the *SYSTIML* register are always zero while the max value of this register is 999,999,999 dec). When synchronized, the *SYSTIM* registers defines the absolute time relative to PTP "epoch" which is January 1st 1970 00:00:00 International Atomic Time (TAI).

- Initial Setting - Setting the initial time is done by direct write access to the *SYSTIM* register. Software should first set the *SYSTIML register* and then set the *SYSTIMH register.* Setting the *SYSTIMR register* is meaningless while it represents sub ns units. It is recommended to disable the timer at programming time as follows:

- Run Time - During run time the *SYSTIM* timer value in the *SYSTIMH*, *SYSTIML* and *SYSTIMR* registers, is updated periodically each 8 nS clock cycle according to the following formula:
  - Define: INC_TIME = 8 nsec +/- TIMINCA.Incvalue * 2-32 nsec. Add or subtract the *TIMINCA.Incvalue* is defined by *TIMINCA.ISGN* (while 0b means Add and 1b means Subtract)
  - Then: SYSTIM = SYSTIM + INC_TIME
- Reading the SYSTIM register by software is done by the following sequence:
  - Read the *SYSTIMR*register
  - Read the *SYSTIML*register
  - Read the *SYSTIMH*register
- Dynamic update of SYSTIM registers can be done by using the TIMADJ registers by the following flow. It can also be done by adjusting the INC_TIME as described later in this section. Adjusting the time by TIMADJ are meant to be used only when the time difference between the master and the slave are small enough (at least smaller than one 8th of the time between consecutive SYNC cycles). If this assumption is incorrect, than this process might take longer time than the SYNC cycle to take effect. If this is an issue, software might need to set the SYSTIM by direct access as described in the "Initial Setting" phase:
  - Write the *Tadjust* value and its *Sign* to the *TIMADJ* register (the *Sign* bit indicates if the Tadjust value should be added or subtracted)
  - Following the write access to the *TIMADJ* register, the hardware repeats the following two steps at each 8 nsec clock as long as the Tadjust > zero.
    - SYSTIM = SYSTIM + INC_TIME +/- 1 nsec. Add or subtract 1 nsec is defined by *TIMADJ.Sign* (while 0bmeans Add and 1b means Subtract)
    - Tadjust = Tadjust - 1 nsec
    - Note that the SYSTIM timer is incremented monotonically at all times. When updating the SYSTIM by the TIMADJ and concurrent non-zero TIMINCA, the SYSTIM is incremented each clock by steps in the range of 6.5ns up to 9.5ns units.
  - As shown above, the time adjustment might take multiple clocks. Software might write a new value to the TIMADJ register before the hardware completed the previous adjustment. In such a case, the new value written by software, overrides the above equation. If such a race is not desired, the software could check that the previous adjustment is completed by one of the following methods:
    - Wait enough time before accessing the *TIMADJ* register which guarantees that the previous update procedure is completed.
    - Poll the matched *TSICR.TADJ* flag which is set by the hardware each time the update procedure is completed.
    - Enable the TADJ interrupt by setting the TADJ flag in the TSIM register and enable timesync interrupts by setting the Time_Sync flag in the IMS register. The TADJ interrupt indicates that the hardware completed the adjustment procedure. This method is unlikely to be used in nominal operation since the expected adjustments are in the sub μs range.
- Dynamic update of SYSTIM registers can also be done by updating the INC_TIME. Using INC_TIME, the time in the slave is updated in a more gradual manner and in most cases it results in a more accurate timer. INC_TIME should be updated as a function of the required Tadjust and the time gap between consecutive SYNC cycles that generated this Tadjust value. A possible equation for the INC_TIME for the next SYNC cycle can be as follows:

  INC_TIME (n+1) = INC_TIME (n) * (T4-T1)/(T3-T2) + Factor * (T4 - (T3 + Tdelay)) / Tcycle

  while
  - All time parameters are expressed in ns units
  - INC_TIME = 8 +/- TIMINCA.Incvalue * 2-32 [ns]
  - INC_TIME (n) and INC_TIME (n+1) are the INC_TIME used for the current Tcycle and the calculated INC_TIME that should be used in the next Tcycle respectively
  - Tcycle is the time between consecutive (Sync request + Delay request) cycles



— Tdelay is the transmission delay from the slave to the master. It can be calculated using T1...T4 as follow: Tdelay = [(T2-T1) + (T4-T3)] / 2

— The factor is a parameter that affects the speed of convergence. For a clock frequency of 125 MHz, an optimized factor equals 8. Table 6-64 lists the expected convergence time for some cases while Tcycle equals 1 second and the slave-to-master clock frequency difference equals 100 ppm.

6.8.3.3 Target Time

The two target time registers *TRGTTIML/H0* and *TRGTTIML/H1* enable generating a time triggered event to external hardware using one of the SDP pins according to the setup defined in the *TSSDP* and *TSAUXC* registers (See Section 7.15.13 and Section 7.15.25). Each target time register is structured the same as the *SYSTIML/H* registers. If the value of *SYSTIML/H* is equal or larger than the value of the *TRGTTIML/H* registers, a change in level or a pulse is generated on the matched SDP outputs.

6.8.3.3.1 SYSTIM Synchronized Level Change Generation on SDP Pins

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. 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 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.
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.
4. Program the target time *TRGTTIML/Hx* (while 'x' is 0b or 1b) to the required event time.
5. 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).
6. When the *SYSTIML/H* registers becomes equal or larger than the selected *TRGTTIML/H* registers, the selected SDP changes its output level.

6.8.3.3.2 SYSTIM Synchronized Pulse Generation on SDP Pins

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. 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 a specific SDP pin by setting the *TSSDP.TS_SDPx_EN* flag to 1b (while 'x' is 0, 1, 2 or 3).
2. 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.
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.
4. 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.
5. 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).

6. When the *SYSTIML/H* registers becomes equal or larger than the *TRGTTIML/H* registers that define the beginning of the pulse, the selected SDP changes its level. Then, when the *SYSTIML/H* registers becomes equal or larger than the other *TRGTTIML/H* registers (that define the trailing edge of the pulse), the selected SDP changes its level back.

6.8.3.3.3 Synchronized Output Clock on SDP Pins

The I210-CS/CL 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 a specific SDP pin by setting the *TSSDP.TS_SDPx_EN* flag to 1b (while 'x' is 0, 1, 2 or 3).
2. 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.
3. 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 6.8.3.3.1. 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 (as described in bullet 5 below)
   b. 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)
4. 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.
5. TRGTTIML/Hx should be set to the required start time of the low phase of the clock.
6. 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 reaches TRGTTIM, hardware begins an endless loop of the following two steps:

1. Increment the used *TRGTTIML/Hx* by *FREQOUT*.
2. When *SYSTIM* is equal or larger than the *TRGTTIM*, the SDP reverts its output level.

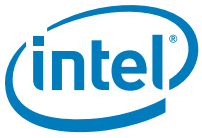
6.8.3.4 Time Stamp Events

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*). Software enables the timestamp of input event as follow:

1. Define the sampled SDP on AUX time '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".

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.

6.8.4 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 7.16. Occurrence of a Time Sync interrupt sets the *ICR.Time_Sync* interrupt bit.

6.8.5 PTP Packet Structure

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 6-63 and Table 6-64. The PTP V1 and V2 message formats are listed in the Table 6-65 followed by SYNC packet format in Table 6-66. Table 6-67 and Table 6-68 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 6-69 lists the device settings required to identify the PTP packets.

Table 6-63. PTP Message Over Layer 2

Ethernet (L2)	VLAN (Optional)	PTP Ethertype	PTP message

Table 6-64. PTP Message Over Layer 4

Ethernet (L2)	IP (L3)	UDP	PTP message

Table 6-65. 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			

Table 6-65. V1 and V2 PTP Message Header (Continued)

Offset in Bytes	V1 Fields	V2 Fields
Bits	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
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 6-66. SYNC Message Structure

Offset in Bytes	Length in Bytes	Fields
0	34	Message Header (as listed in Table 6-65).
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

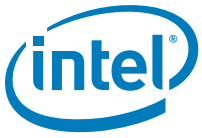


Table 6-67. Message Decoding for V1 (Control Field at Offset 32)

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 6-68. 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

The I210-CS/CL identifies both L2 and L4 PTP packets for timestamp sampling and defining a specific receive queue as listed in the Table 6-69.

Table 6-69. 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.
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)

Table 6-69. Enabling Receive Timestamp

Functionality	Register	Field	Setting Options
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.

6.9 Statistic Counters

The I210-CS/CL supports different statistic counters as described in Section 7.18. The statistic counters can be used to create statistic reports as required by different standards. The I210-CS/CL 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

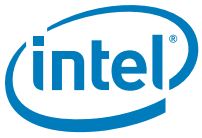
The following section describes the match between the internal the I210-CS/CL statistic counters and the counters requested by the different standards.

6.9.1 IEEE 802.3 Clause 30 Management

The I210-CS/CL supports the Basic and Mandatory Packages defined in clause 30 of the IEEE 802.3 specification. Table 6-70 lists the matching between the internal statistics and the counters requested by these packages.

Table 6-70. IEEE 802.3 Mandatory Package Statistics

Mandatory Package Capability	I210-CS/CL Counter	Notes and Limitations
FramesTransmittedOK	GPTC	The I210-CS/CL doesn't include flow control packets.
SingleCollisionFrames	SCC	
MultipleCollisionFrames	MCC	
FramesReceivedOK	GPRC	The I210-CS/CL doesn't include flow control packets.
FrameCheckSequenceErrors	CRCERRS	
AlignmentErrors	ALGNERRC	



In addition, part of the recommended package is also implemented as listed in Table 6-71.

Table 6-71. IEEE 802.3 Recommended Package Statistics

Recommended package capability	I210-CS/CL Counter	Notes and Limitations
OctetsTransmittedOK	GOTCH/GOTCL	The I210-CS/CL counts also the DA/SA/LT/CRC as part of the octets. The I210-CS/CL doesn't count Flow control packets.
FramesWithDeferredXmissions	DC	
LateCollisions	LATECOL	
FramesAbortedDueToXSColls	ECOL	
FramesLostDueToIntMACXmitError	HTDMPC	The I210-CS/CL counts the excessive collisions in this counter, while 802.3 increments no other counters, while this counter is incremented
CarrierSenseErrors	TNCRS	The I210-CS/CL 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 The I210-CS/CL this counter is not operational in 100 Mbps half duplex mode.
OctetsReceivedOK	TORL+TORH	The I210-CS/CL 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 6-75.

Table 6-72. IEEE 802.3 Optional Package Statistics

Optional package capability	I210-CS/CL Counter	Notes
MulticastFramesXmittedOK	MPTC	The I210-CS/CL doesn't count FC packets
BroadcastFramesXmittedOK	BPTC	
MulticastFramesReceivedOK	MPRC	The I210-CS/CL doesn't count FC packets
BroadcastFramesReceivedOK	BPRC	
InRangeLengthErrors	LENERRS	
OutOfRangeLengthField	N/A	Packets parsed as Ethernet II packets
FrameTooLongErrors	ROC + RJC	

6.9.2 OID_GEN_STATISTICS

The I210-CS/CL supports the part of the OID_GEN_STATISTICS as defined by Microsoft* NDIS 6.0 specification. Table 6-73 lists the matching between the internal statistics and the counters requested by this structure.

Table 6-73. Microsoft* OID_GEN_STATISTICS

OID entry	I210-CS/CL Counters	Notes
ifInDiscards;	CRCERRS + RLEC + RXERRC + MPC + RNBC + ALGNERRC	
ifInErrors;	CRCERRS + RLEC + RXERRC + ALGNERRC	
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	

6.9.3 RMON

The I210-CS/CL supports the part of the RMON Ethernet statistics group as defined by IETF RFC 2819. Table 6-74 lists the matching between the internal statistics and the counters requested by this group.

Table 6-74. RMON Statistics

RMON statistic	I210-CS/CL Counters	Notes
etherStatsDropEvents	*MPC + RNBC*	
etherStatsOctets	*TOTL + TOTH*	
etherStatsPkts	TPR	
etherStatsBroadcastPkts	BPRC	
etherStatsMulticastPkts	MPRC	The I210-CS/CL 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

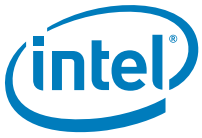


Table 6-74. RMON Statistics (Continued)

RMON statistic	I210-CS/CL Counters	Notes
etherStatsPkts256to511Octets	PRC511	RMON counts bad packets as well
etherStatsPkts512to1023Octets	PRC1023	RMON counts bad packets as well
etherStatsPkts1024to1518Octets	PRC1522	RMON counts bad packets as well

6.9.4 Linux net_device_stats

The I210-CS/CL supports part of the net_device_stats as defined by Linux Kernel version 2.6 (defined in <linux/netdevice.h>). Table 6-75 lists the matching between the internal statistics and the counters requested by this structure.

Table 6-75. Linux net_device_stats

net_device_stats field	I210-CS/CL Counters	Notes
rx_packets	GPRC	The I210-CS/CL doesn't count flow controls - can be accounted for by using the XONRXC and XOFFRXC counters
tx_packets	GPTC	The I210-CS/CL 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	
tx_heartbeat_errors	N/A	
tx_window_errors	LATECOL	
rx_compressed	N/A	
tx_compressed	N/A	

6.9.5 Statistics Hierarchy

The following diagram shows the relations between the packet flow and the different statistic counters.

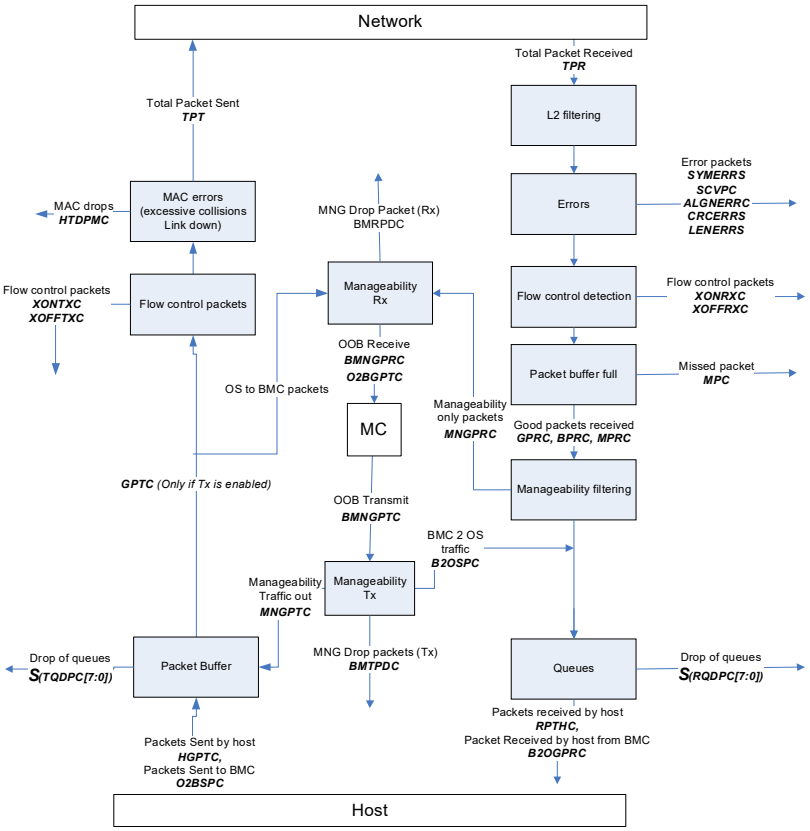
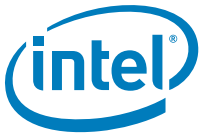


Figure 6-21. Flow Statistics



7.0 Programming Interface

7.1 Introduction

This section details the programmer visible state inside the I210-CS/CL. In some cases, it describes hardware structures invisible to software in order to clarify a concept. The I210-CS/CL's address space is mapped into four regions with PCI Base Address registers described in Section 8.3.11. These regions are listed in Table 7-1.

Table 7-1. 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.

7.1.1 Memory, I/O Address and Configuration Decoding

7.1.1.1 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 8.3.11). Refer to Section 7.1.3 for the appropriate offset for each specific internal register.

7.1.1.2 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 8.3.11). 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 Flash.

7.1.1.3 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 8.3.11). Refer to Section 7.1.3 for the appropriate offset for each specific internal MSI-X register.

7.1.1.4 Memory-Mapped Access to Expansion ROM

The Expansion/Option ROM module located in the external Flash 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.

7.1.1.5 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 8.3.11), the BAR contains a valid (non-zero value), and I/O address decoding is enabled in the PCIe configuration.

When an I/O BAR is mapped, the I/O address range allocated opens a 32-byte window in the system I/O address map. Within this window, two I/O addressable registers are implemented: IOADDR and IODATA. The IOADDR register is used to specify a reference to an internal register or memory, and then the IODATA register is used as a window to the register or memory address specified by IOADDR:

Table 7-2. 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 ,Äì 0x1F	Reserved	Reserved.	RO	4 bytes

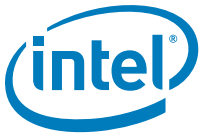
7.1.1.5.1 IOADDR (I/O Offset 0x00)

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.



7.1.1.5.2 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, the I210-CS/CL 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.

7.1.1.5.3 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.

7.1.1.6 Configuration Access to Internal Registers and Memories

To support legacy pre-boot 16-bit operating environments without requiring I/O address space, the I210-CS/CL 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 7-3.

Note: To enable CSR access via configuration address space the *CSR_conf_en* Flash bit should be set.

Table 7-3. 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 Flash 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.

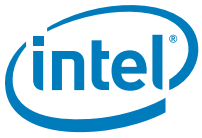
— Software should access CSRs via I/O address space or configuration address space but should not use both mechanisms at the same time.

7.1.2 Register Conventions

All registers in the I210-CS/CL 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 Flash 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 Flash 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.

The I210-CS/CL register fields are assigned one of the attributes listed in Table 7-4.

Table 7-4. I210-CS/CL 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 strapping, serial Flash 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.
R/W1C	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 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 7-4. I210-CS/CL Register Field Attributes (Continued)

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.
R/W1	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 7-5).

Table 7-5. 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.
R/W	Read and Write.
SC	Self-Clear. The bit is set, automatically executed, and then reset to normal operation.
CR	Clear after Read. For example, 1000BASE-T Status Register bits 7:0 (Idle Error Counter).
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.

7.1.2.1 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):

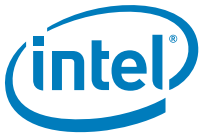
```
Last                          First
...,06, 05, 04, 03, 02, 01, 00
```

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, the EType field in the METF 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
```

7.1.3 Register Summary

All the I210-CS/CL's non-PCIe configuration registers, except for the MSI-X register, are listed in Table 7-6. These registers are ordered by grouping and are not necessarily listed in order that they appear in the address space.

Table 7-6. Register Summary

Offset	Alias Offset	Abbreviation	Name	RW
General				
0x0000	0x0004	CTRL	Device Control Register	RW
0x0008	N/A	STATUS	Device Status Register	RO
0x0018	N/A	CTRL_EXT	Extended Device Control Register	RW
0x0020	N/A	MDIC	MDI Control Register	RW
0x0028	N/A	FCAL	Flow Control Address Low	RO
0x002C	N/A	FCAH	Flow Control Address High	RO
0x0030	N/A	FCT	Flow Control Type	RW
0x0034	N/A	CONNSW	Copper/Fiber Switch Control	RW
0x0038	N/A	VET	VLAN Ether Type	RW
0x0E04	N/A	MDICNFG	MDC/MDIO Configuration Register	RW
0x0170	N/A	FCTTV	Flow Control Transmit Timer Value	RW
0x0E00	N/A	LEDCTL	LED Control Register	RW
0x1028	N/A	I2CCMD	SFP I^2C Command	RW
0x102C	N/A	I2CPARAMS	SFP I^2C Parameter	RW
0x1040	N/A	WDSTP	Watchdog Setup Register	RW
0x1044	N/A	WDSWSTS	Watchdog Software	RW
0x1048	N/A	FRTIMER	Free Running Timer	RWM

Table 7-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x104C	N/A	TCPTimer	TCP Timer	RW
0x5B70	N/A	DCA_ID	DCA Requester ID Information Register	RO
0x5B50	N/A	SWSM	Software Semaphore Register	RW
0x5B54	N/A	FWSM	Firmware Semaphore Register	RWM
0x5B5C	N/A	SW_FW_SYNC	Software-Firmware Synchronization	RWM
0x0E38	N/A	IPCNFG	Internal PHY Configuration	RW
0x0E14	N/A	PHPM	PHY Power Management	RW
Flash-Security Registers				
0x12010	0x00010	EEC	EEPROM-Mode Control Register	RW
0x12014	0x00014	EERD	EEPROM-Mode Read Register	RW
0x12018	0x1101C	EEWR	EEPROM-Mode Write Register	RW
0x1201C	0x0001C	FLA	Flash Access Register	RW
0x12024	N/A	EEARBC	EEPROM Block Auto Read Bus Control	RW
0x12000	N/A	FLASHMODE	Flash Mode Register	RO
0x12054	N/A	FLASHOP	Flash OP-Code Register	RO
0x12058	N/A	FLASHGOP	Flash General Purpose OP-Code Register	RO
0x12004	N/A	FLASHTIME	Flash Access Timing Register	RO
0x12100	N/A	FLBLKBASE	Flash Block Base Address	RO
0x12104	N/A	FLBLKEND	Flash Block End Address	RO
0x12108	N/A	FLFWUPDATE	Flash Firmware Code Update	RW
0x1210C	N/A	EEBLKBASE	EEPROM Block Base Address	RO
0x12110	N/A	EEBLKEND	EEPROM Block End Address	RO
0x12048	N/A	FLSWCTL	Software Flash Burst Control Register	RW
0x1204C	N/A	FLSWDATA	Software Flash Burst Data Register	RW
0x12050	N/A	FLSWCNT	Software Flash Burst Access Counter	RW
0x12120 - 0x1221C	N/A	INVM_DATA[0 - 63]	iNVM Data Register	R/W1
0x12220 - 0x1229C	N/A	INVM_LOCK[0 -31]	iNVM Lock Register	R/W1
0x12324	N/A	INVM_PROTECT	iNVM Protect Register	RW
Interrupts				
0x1500	0x00C0	ICR	Interrupt Cause Read	RC/W1C
0x1504	0x00C8	ICS	Interrupt Cause Set	WO
0x1508	0x00D0	IMS	Interrupt Mask Set/Read	RW
0x150C	0x00D8	IMC	Interrupt Mask Clear	WO
0x1510	0x00E0	IAM	Interrupt Acknowledge Auto Mask	RW
0x1520	N/A	EICS	Extended Interrupt Cause Set	WO
0x1524	N/A	EIMS	Extended Interrupt Mask Set/Read	RWM
0x1528	N/A	EIMC	Extended Interrupt Mask Clear	WO
0x152C	N/A	EIAC	Extended Interrupt Auto Clear	RW
0x1530	N/A	EIAM	Extended Interrupt Auto Mask	RW
0x1580	N/A	EICR	Extended Interrupt Cause Read	RC/W1C
0x1700 - 0x170C	N/A	IVAR	Interrupt Vector Allocation Registers	RW

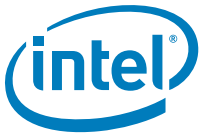


Table 7-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x1740	N/A	IVAR_MISC	Interrupt Vector Allocation Registers - MISC	RW
0x1680 - 0x16A0	N/A	EITR	Extended Interrupt Throttling Rate 0 - 4	RW
0x1514	N/A	GPIE	General Purpose Interrupt Enable	RW
0x5B68	N/A	PBACL	MSI-X PBA Clear	R/W1C
Receive				
0x0100	N/A	RCTL	Rx Control	RW
0x2160	0x0168	FCRTL0	Flow Control Receive Threshold Low	RW
0x2168	0x0160	FCRTH0	Flow Control Receive Threshold High	RW
0x2404	N/A	RXPBSIZE	Rx Packet Buffer Size	RW
0x2460	N/A	FCRTV	Flow Control Refresh Timer Value	RW
0xC000	0x0110, 0x2800	RDBAL[0]	Rx Descriptor Base Low Queue 0	RW
0xC004	0x0114, 0x2804	RDBAH[0]	Rx Descriptor Base High Queue 0	RW
0xC008	0x0118, 0x2808	RDLEN[0]	Rx Descriptor Ring Length Queue 0	RW
0xC00C	0x280C	SRRCTL[0]	Split and Replication Receive Control Register Queue 0	RW
0xC010	0x0120, 0x2810	RDH[0]	Rx Descriptor Head Queue 0	RO
0xC018	0x0128, 0x2818	RDT[0]	Rx Descriptor Tail Queue 0	RW
0xC028	0x02828	RXDCTL[0]	Receive Descriptor Control Queue 0	RW
0xC014	0x2814	RXCTL[0]	Receive Queue 0 DCA CTRL Register	RW
0xC040 + 0x40 * (n-1)	0x2900+ 0x100 * (n-1)	RDBAL[1 - 3]	Rx Descriptor Base Low Queue 1 - 3	RW
0xC044 + 0x40 * (n-1)	0x2904 + 0x100 * (n-1)	RDBAH[1 - 3]	Rx Descriptor Base High Queue 1 - 3	RW
0xC048 + 0x40 * (n-1)	0x2908 + 0x100 * (n-1)	RDLEN[1 - 3]	Rx Descriptor Ring Length Queue 1 - 3	RW
0xC04C + 0x40 * (n-1)	0x290C + 0x100 * (n-1)	SRRCTL[1 - 3]	Split and Replication Receive Control Register Queue 1 - 3	RW
0xC050 + 0x40 * (n-1)	0x2910 + 0x100 * (n-1)	RDH[1 - 3]	Rx Descriptor Head Queue 1 - 3	RO
0xC058 + 0x40 * (n-1)	0x2918 + 0x100 * (n-1)	RDT[1 - 3]	Rx Descriptor Tail Queue 1 - 3	RW
0xC068 + 0x40 * (n-1)	0x2928 + 0x100 * (n-1)	RXDCTL[1 - 3]	Receive Descriptor Control Queue 1 - 3	RW
0xC054 + 0x40 * (n-1)	0x2914 + 0x100 * (n-1)	RXCTL[1 - 3]	Receive Queue 1 - 3 DCA CTRL Register	RW
0x5000	N/A	RXCSUM	Receive Checksum Control	RW
0x5004	N/A	RLPML	Receive Long packet maximal length	RW
0x5008	N/A	RFCTL	Receive Filter Control Register	RW

Table 7-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x5200- 0x53FC	0x0200-0x03FC	MTA[127:0]	Multicast Table Array (n)	RW
0x5400 + 8*n	0x0040 + 8*n	RAL[0-15]	Receive Address Low (15:0)	RW
0x5404 + 8 *n	0x0044 + 8 *n	RAH[0-15]	Receive Address High (15:0)	RW
0x5480 – 0x549C	N/A	PSRTYPE[3:0]	Packet Split Receive type (n)	RW
0x5600-0x57FC	0x0600-0x07FC	VFTA[127:0]	VLAN Filter Table Array (n)	RW
0x5818	N/A	MRQC	Multiple Receive Queues Command	RW
0x5C00-0x5C7C	N/A	RETA	Redirection Table	RW
0x5C80-0x5CA4	N/A	RSSRK	RSS Random Key Register	RW
0xC038 + 0x40*n	N/A	DVMOLR[0 - 3]	DMA VM Offload Register[0-3]	RW
Transmit				
0x0400	N/A	TCTL	Tx Control	RW
0x0404	N/A	TCTL_EXT	Tx Control Extended	RW
0x0410	N/A	TIPG	Tx IPG	RW
0x041C	N/A	RETX_CTL	Retry Buffer Control	RW
0x3404	N/A	TXPBSIZE	Transmit Packet Buffer Size	RW
0x359C	N/A	DTXTCPFLGL	DMA Tx TCP Flags Control Low	RW
0x35A0	N/A	DTXTCPFLGH	DMA Tx TCP Flags Control High	RW
0x3540	N/A	DTXMXSZRQ	DMA Tx Max Total Allow Size Requests	RW
0x355C	N/A	DTXMXPKTSZ	DMA Tx Max Allowable Packet Size	RW
0x3590	N/A	DTXCTL	DMA Tx Control	RW
0x35A4	N/A	DTXBCTL	DMA Tx Behavior Control	RW
0xE000	0x0420, 0x3800	TDBAL[0]	Tx Descriptor Base Low 0	RW
0xE004	0x0424, 0x3804	TDBAH[0]	Tx Descriptor Base High 0	RW
0xE008	0x0428, 0x3808	TDLEN[0]	Tx Descriptor Ring Length 0	RW
0xE010	0x0430, 0x3810	TDH[0]	Tx Descriptor Head 0	RO
0xE018	0x0438, 0x3818	TDT[0]	Tx Descriptor Tail 0	RW
0xE028	0x3828	TXDCTL[0]	Transmit Descriptor Control Queue 0	RW
0xE014	0x3814	TXCTL[0]	Tx DCA CTRL Register Queue 0	RW
0xE038	0x3838	TDWBAL[0]	Transmit Descriptor WB Address Low Queue 0	RW
0xE03C	0x383C	TDWBAH[0]	Transmit Descriptor WB Address High Queue 0	RW
0xE040 + 0x40 * (n-1)	0x3900 + 0x100 * (n-1)	TDBAL[1-3]	Tx Descriptor Base Low Queue 1 - 3	RW
0xE044 + 0x40 * (n-1)	0x3904 + 0x100 * (n-1)	TDBAH[1-3]	Tx Descriptor Base High Queue 1 - 3	RW
0xE048 + 0x40 * (n-1)	0x3908 + 0x100 * (n-1)	TDLEN[1-3]	Tx Descriptor Ring Length Queue 1 - 3	RW

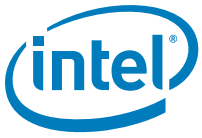


Table 7-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0xE050 + 0x40 * (n-1)	0x3910 + 0x100 * (n-1)	TDH[1-3]	Tx Descriptor Head Queue 1 - 3	RO
0xE058 + 0x40 * (n-1)	0x3918 + 0x100 * (n-1)	TDT[1-3]	Tx Descriptor Tail Queue 1 - 3	RW
0xE068 + 0x40 * (n-1)	0x3928 + 0x100 * (n-1)	TXDCTL[1-3]	Transmit Descriptor Control 1 - 3	RW
0xE054 + 0x40 * (n-1)	0x3914 + 0x100 * (n-1)	TXCTL[1-3]	Tx DCA CTRL Register Queue 1 - 3	RW
0xE078 + 0x40 * (n-1)	0x3938 + 0x100 * (n-1)	TDWBAL[1-3]	Transmit Descriptor WB Address Low Queue 1 - 3	RW
0xE07C + 0x40 * (n-1)	0x393C + 0x100 * (n-1)	TDWBAH[1-3]	Transmit Descriptor WB Address High Queue 1 - 3	RW
0x300C + 0x40 *n	N/A	TQAVHC	Transmit Qav High Credits	RW
0x3004 + 0x40*n	N/A	TQAVCC[0-1]	Transmit Qav	RW
0x3570	N/A	TQAVCTRL	Transmit Qav Control	RW
Filters				
0x5CB0 + 4*n	N/A	ETQF[0 - 7]	EType Queue Filter 0 - 7	RW
0x5A80 + 4*n	N/A	IMIR[0 - 7]	Immediate Interrupt Rx 0 - 7	RW
0x5AA0 + 4*n	N/A	IMIREXT[0 - 7]	Immediate Interrupt Rx Extended 0 - 7	RW
0x5AC0	N/A	IMIRVP	Immediate Interrupt Rx VLAN Priority	RW
0x59E0 + 4*n	N/A	TTQF[0 - 7]	Two-Tuple Queue Filter 0 - 7	RW
0x55FC	N/A	SYNQF	SYN Packet Queue Filter	RW
Per Queue Statistics				
0xC030 + 0x40 * n	0x2830 + 0x100 * n	RQDPC[0 - 3]	Receive Queue Drop Packet Count Register 0 - 3	RW
0xE030 + 0x40 * n	N/A	TQDPC[0 - 3]	Transmit Queue Drop Packet Count Register 0 - 3	RW
0x10010 + 0x100*n	N/A	PQGPRC[0 - 3]	Per Queue Good Packets Received Count	RO
0x10014 + 0x100*n	N/A	PQGPTC[0 - 3]	Per Queue Good Packets Transmitted Count	RO
0x10018 + 0x100*n	N/A	PQGORC[0 - 3]	Per Queue Good Octets Received Count	RO
0x10034 + 0x100*n	N/A	PQGOTC[0 - 3]	Per Queue Octets Transmitted Count	RO
0x10038 + 0x100*n	N/A	PQMPRC[0 - 3]	Per Queue Multicast Packets Received Count	RO
Statistics				
0x4000	N/A	CRCERRS	CRC Error Count	RC
0x4004	N/A	ALGNERRC	Alignment Error Count	RC
0x4008	N/A	SYMERRS	Symbol Error Count	RC
0x400C	N/A	RXERRC	Rx Error Count	RC
0x4010	N/A	MPC	Missed Packets Count	RC

Table 7-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x4014	N/A	SCC	Single Collision Count	RC
0x4018	N/A	ECOL	Excessive Collisions Count	RC
0x401C	N/A	MCC	Multiple Collision Count	RC
0x4020	N/A	LATECOL	Late Collisions Count	RC
0x4028	N/A	COLC	Collision Count	RC
0x4030	N/A	DC	Defer Count	RC
0x4034	N/A	TNCRS	Transmit - No CRS	RC
0x403C	N/A	HTDPMC	Host Transmit Discarded Packets by MAC Count	RC
0x4040	N/A	RLEC	Receive Length Error Count	RC
0x4048	N/A	XONRXC	XON Received Count	RC
0x404C	N/A	XONTXC	XON Transmitted Count	RC
0x4050	N/A	XOFFRXC	XOFF Received Count	RC
0x4054	N/A	XOFFTXC	XOFF Transmitted Count	RC
0x4058	N/A	FCRUC	FC Received Unsupported Count	RC
0x405C	N/A	PRC64	Packets Received (64 Bytes) Count	RC
0x4060	N/A	PRC127	Packets Received (65-127 Bytes) Count	RC
0x4064	N/A	PRC255	Packets Received (128-255 Bytes) Count	RC
0x4068	N/A	PRC511	Packets Received (256-511 Bytes) Count	RC
0x406C	N/A	PRC1023	Packets Received (512-1023 Bytes) Count	RC
0x4070	N/A	PRC1522	Packets Received (1024-1522 Bytes)	RC
0x4074	N/A	GPRC	Good Packets Received Count	RC
0x4078	N/A	BPRC	Broadcast Packets Received Count	RC
0x407C	N/A	MPRC	Multicast Packets Received Count	RC
0x4080	N/A	GPTC	Good Packets Transmitted Count	RC
0x4088	N/A	GORCL	Good Octets Received Count (Lo)	RC
0x408C	N/A	GORCH	Good Octets Received Count (Hi)	RC
0x4090	N/A	GOTCL	Good Octets Transmitted Count (Lo)	RC
0x4094	N/A	GOTCH	Good Octets Transmitted Count (Hi)	RC
0x40A0	N/A	RNBC	Receive No Buffers Count	RC
0x40A4	N/A	RUC	Receive Under Size Count	RC
0x40A8	N/A	RFC	Receive Fragment Count	RC
0x40AC	N/A	ROC	Receive Oversize Count	RC
0x40B0	N/A	RJC	Receive Jabber Count	RC
0x40B4	N/A	MNGPRC	Management Packets Receive Count	RC
0x40B8	N/A	MPDC	Management Packets Dropped Count	RC
0x40BC	N/A	MNGPTC	Management Packets Transmitted Count	RC
0x40C0	N/A	TORL	Total Octets Received (Lo)	RC
0x8FE0	N/A	B2OSPC	BMC2OS Packets Sent by MC	RC
0x4158	N/A	B2OGPRC	BMC2OS Packets Received by Host	RC
0x8FE4	N/A	O2BGPTC	OS2BMC Packets Received by MC	RC
0x415C	N/A	O2BSPC	OS2BMC Packets Transmitted By Host	RC
0x40C4	N/A	TORH	Total Octets Received (Hi)	RC
0x40C8	N/A	TOTL	Total Octets Transmitted (Lo)	RC
0x40CC	N/A	TOTH	Total Octets Transmitted (Hi)	RC

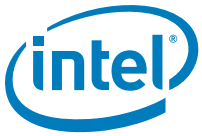


Table 7-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x40D0	N/A	TPR	Total Packets Received	RC
0x40D4	N/A	TPT	Total Packets Transmitted	RC
0x40D8	N/A	PTC64	Packets Transmitted (64 Bytes) Count	RC
0x40DC	N/A	PTC127	Packets Transmitted (65-127 Bytes) Count	RC
0x40E0	N/A	PTC255	Packets Transmitted (128-256 Bytes) Count	RC
0x40E4	N/A	PTC511	Packets Transmitted (256-511 Bytes) Count	RC
0x40E8	N/A	PTC1023	Packets Transmitted (512-1023 Bytes) Count	RC
0x40EC	N/A	PTC1522	Packets Transmitted (1024-1522 Bytes) Count	RC
0x40F0	N/A	MPTC	Multicast Packets Transmitted Count	RC
0x40F4	N/A	BPTC	Broadcast Packets Transmitted Count	RC
0x40F8	N/A	TSCTC	TCP Segmentation Context Transmitted Count	RC
0x4100	N/A	IAC	Interrupt Assertion Count	RC
0x4104	N/A	RPTHC	Rx Packets to Host Count	RC
0x4118	N/A	HGPTC	Host Good Packets Transmitted Count	RC
0x4120	N/A	RXDMTC	Rx Descriptor Minimum Threshold Count	RC
0x4128	N/A	HGORCL	Host Good Octets Received Count (Lo)	RC
0x412C	N/A	HGORCH	Host Good Octets Received Count (Hi)	RC
0x4130	N/A	HGOTCL	Host Good Octets Transmitted Count (Lo)	RC
0x4134	N/A	HGOTCH	Host Good Octets Transmitted Count (Hi)	RC
0x4138	N/A	LENERRS	Length Errors Count Register	RC
0x4228	N/A	SCVPC	SerDes/SGMII/1000BASE-KX Code Violation Packet Count Register	RW
Wake Up and Proxying				
0x5800	N/A	WUC	Wake Up Control	RW
0x5808	N/A	WUFC	Wake Up Filter Control	RW
0x5810	N/A	WUS	Wake Up Status	R/W1C
0x5F60	N/A	PROXYFC	Proxying Filter Control	RW
0x5F64	N/A	PROXYS	Proxying Status	R/W1C
0x5838	N/A	IPAV	IP Address Valid	RW
0x5840- 0x5858	N/A	IP4AT	IPv4 Address Table	RW
0x5880- 0x588F	N/A	IP6AT	IPv6 Address Table	RW
0x5900	N/A	WUPL	Wake Up Packet Length	RO
0x5A00- 0x5A7C	N/A	WUPM	Wake Up Packet Memory	RO
0x9000-0x93FC	N/A	FHFT	Flexible Host Filter Table Registers	RW
0x9A00-0x9DFC	N/A	FHFT_EXT	Flexible Host Filter Table Registers Extended	RW
0x5590	N/A	PROXYFCEX	Proxy Filter Control Extended	RW
0x5594	N/A	PROXYEXS	Proxy Extended Status	R/W1C
0x5500-0x557C	N/A	WFUTPF[31:0]	Wake Flex UDP TCP Port Filter	RW
0x5580	N/A	RFUTPF	Range Flex UDP TCP Port Filter	RW
0x5584	N/A	RWPFC	Range Wake Port Filter Control	RW
0x5588	N/A	WFUTPS	Wake Filter UDP TCP Status	R/W1C
0x558C	N/A	WCS	Wake Control Status	R/W1C

Table 7-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
PCIe				
0x5B00	N/A	GCR	PCIe Control Register	RW
0x5B10	N/A	GSCL_1	PCIe Statistics Control #1	RW
0x5B14	N/A	GSCL_2	PCIe Statistics Control #2	RW
0x5B90 - 0x5B9C	N/A	GSCL_5_8	PCIe Statistics Control Leaky Bucket Timer	RW
0x5B20	N/A	GSCN_0	PCIe Counter Register #0	RW
0x5B24	N/A	GSCN_1	PCIe Counter Register #1	RW
0x5B28	N/A	GSCN_2	PCIe Counter Register #2	RW
0x5B2C	N/A	GSCN_3	PCIe Counter Register #3	RW
0x5B30	N/A	FACTPS	Function Active and Power State	RW
0x5B64	N/A	MREVID	Mirrored Revision ID	RO
0x5B6C	N/A	GCR_EXT	PCIe Control Extended Register	RW
0x5B74	N/A	DCA_CTRL	DCA Control Register	RW
0x5B88	N/A	PICAUSE	PCIe Interrupt Cause	R/W1C
0x5B8C	N/A	PIENA	PCIe Interrupt Enable	RW
0x5BFC	N/A	BARCTRL	PCIe BAR Control	RW
0x5BF4	N/A	RR2DCDELAY	Read Request To Data Completion Delay Register	RC
0x5B4C	N/A	PCIEMCTP	PCIe MCTP Register	RW to FW
Memory Error Detection				
0x1084	N/A	PEIND	Parity and ECC Indication	RC
0x1088	N/A	PEINDM	Parity and ECC Indication Mask	RW
0x245C	N/A	PBECCSTS	Packet Buffer ECC Status	RW
0x5BA0	N/A	PCIEERRCTL	PCIe Parity Control Register	RW
0x5BA4	N/A	PCIEECCCTL	PCIe ECC Control Register	RW
0x5BA8	N/A	PCIEERRSTS	PCIe Parity Status Register	R/W1C
0x5BAC	N/A	PCIEECCSTS	PCIe ECC Status Register	R/W1C
0x5B7C	N/A	PCIEACL01	PCIe ACL0 and ACL1 Register	RW to FW
0x5B80	N/A	PCIEACL23	PCIe ACL2 and ACL3 Register	RW to FW
0x5F54	N/A	LANPERRCTL	LAN Port Parity Error Control Register	RW to FW
0x5F58	N/A	LANPERRSTS	LAN Port Parity Error Status Register	R/W1C
0x5F5C	N/A	LANPERRINJ	LAN Port Parity Error Inject Register	SC
Power Management Registers				
0x2508	N/A	DMACR	DMA Coalescing Control Register	RW
0x2514	N/A	DMCTLX	DMA Coalescing Time to LX Request	RW
0x3550	N/A	DMCTXTH	DMA Coalescing Transmit Threshold	RW
0x5DD4	N/A	DMCCNT	DMA Coalescing Current Rx Count	RO
0x2170	N/A	FCRTC	Flow Control Receive Threshold Coalescing	RW
0x5DC8	N/A	DMACTC	DMA Coalescing Clock Control Time Counter	RO
Diagnostic				
0x5BB8	N/A	PCIEMISC	PCIe Misc. Register	RW
PCS				
0x4200	N/A	PCS_CFG	PCS Configuration 0 Register	RW

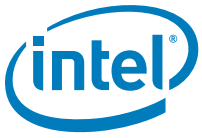


Table 7-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x4208	N/A	PCS_LCTL	PCS Link Control Register	RW
0x420C	N/A	PCS_LSTS	PCS Link Status Register	RO
0x4210	N/A	PCS_DBG0	PCS Debug 0 Register	RO
0x4214	N/A	PCS_DBG1	PCS Debug 1 Register	RO
0x4218	N/A	PCS_ANADV	AN Advertisement Register	RW
0x421C	N/A	PCS_LPAB	Link Partner Ability Register	RO
0x4220	N/A	PCS_NPTX	AN Next Page Transmit Register	RW
0x4224	N/A	PCS_LPABNP	Link Partner Ability Next Page Register	RO
Time Sync				
0xB620	N/A	TSYNCRXCTL	Rx Time Sync Control Register	RW
0xB624	N/A	RXSTMPL	Rx Timestamp Low	RO
0xB628	N/A	RXSTMPH	Rx Timestamp High	RO
0xB614	N/A	TSYNCTXCTL	Tx Time Sync Control Register	RW
0xB618	N/A	TXSTMPL	Tx Timestamp Value Low	RO
0xB61C	N/A	TXSTMPH	Tx Timestamp Value High	RO
0xB6F8	N/A	SYSTIMR	System Time Residue Register	RW
0xB600	N/A	SYSTIML	System Time Register Low	RW
0xB604	N/A	SYSTIMH	System Time Register High	RW
0xB6FC	N/A	SYSTIMTM	System Time Register Tx MS	RW
0xB608	N/A	TIMINCA	Increment Attributes Register	RW
0xB60C	N/A	TIMADJ	Time Adjustment Offset Register	RW
0xB640	N/A	TSAUXC	Auxiliary Control Register	RW
0xB644	N/A	TRGTTIML0	Target Time Register 0 Low	RW
0xB648	N/A	TRGTTIMH0	Target Time Register 0 High	RW
0xB64C	N/A	TRGTTIML1	Target Time Register 1 Low	RW
0xB650	N/A	TRGTTIMH1	Target Time Register 1 High	RW
0xB654	N/A	FREQOUT0	Frequency Out 0 Control Register	RW
0xB658	N/A	FREQOUT1	Frequency Out 1 Control Register	RW
0xB65C	N/A	AUXSTMPL0	Auxiliary Timestamp 0 Register Low	RO
0xB660	N/A	AUXSTMPH0	Auxiliary Timestamp 0 Register High	RO
0xB664	N/A	AUXSTMPL1	Auxiliary Timestamp 1 Register Low	RO
0xB668	N/A	AUXSTMPH1	Auxiliary Timestamp 1 Register High	RO
0x5F50	N/A	TSYNCRXCFG	Time Sync Rx Configuration	RW
0x003C	N/A	TSSDP	Time Sync SDP Configuration Register	RW
0xB66C	N/A	TSICR	Time Sync Interrupt Cause Register	RC/W1C
0xB674	N/A	TSIM	Time Sync Interrupt Mask Register	RW
0x3578	N/A	LAUNCH_OS0	Launch Time Offset Register 0	RW

7.1.3.1 Alias Addresses

Certain registers maintain an alias address designed for backward compatibility with software written for previous GbE controllers. For these registers, the alias address is listed Table 7-6. 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 the I210-CS/CL, use the new address offset.

7.1.4 MSI-X BAR Register Summary

Table 7-7. MSI-X Register Summary

Category	Offset	Abbreviation	Name	RW	Page
MSI-X Table	0x0000 + n*0x10 [n=0...4]	MSIXTADD	MSI–X Table Entry Lower Address	RW	page 296
MSI-X Table	0x0004 + n*0x10 [n=0...4]	MSIXTUADD	MSI–X Table Entry Upper Address	RW	page 296
MSI-X Table	0x0008 + n*0x10 [n=0...4]	MSIXTMSG	MSI–X Table Entry Message	R/W	page 296
MSI-X Table	0x000C + n*0x10 [n=0...4]	MSIXTVCTRL	MSI–X Table Entry Vector Control	R/W	page 297
MSI-X Table	0x02000	MSIXPBA	MSIXPBA Bit Description	RO	page 297

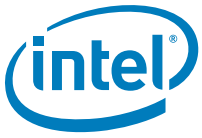
7.2 General Register Descriptions

7.2.1 Device Control Register - CTRL (0x00000; R/W)

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.

Note: This register is also aliased at address 0x0004.

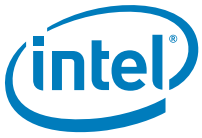
Field	Bit(s)	Initial Value	Description
FD	0	1b[1]	Full-Duplex Controls the MAC duplex setting when explicitly set by software. 0b = Half duplex. 1b = Full duplex.
Reserved	1	0b	Reserved Write 0b; ignore on read.
GIO Master Disable	2	0b	When set to 1b, the function of this bit blocks new master requests. 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 3.6.4 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 Flash then it is initialized to 1b. 2. The *CTRL.SLU* bit is set to 1b if the *Enable All Phys in D3* bit in the Common Firmware Parameters 2 Flash word is set to 1b. 3. The *CTRL.SLU* bit is set in NC-SI mode according to the Enable Channel command to the port. 4. In SerDes and 1000Base-KX modes Link up can be forced by setting this bit as described in Section 3.6.4.1.4.



Field	Bit(s)	Initial Value	Description
ILOS	7	0b[1]	Invert Loss-of-Signal (LOS/LINK) Signal This bit controls the polarity of the SRDS_[n]_SIG_DET signal or internal link-up signal. 0b = Do not invert (active high input signal). 1b = Invert signal (active low input signal). ***Notes:*** 1. Source of the link-up signal (SRDS_[n]_SIG_DET signal or internal link-up signal) is set via the *CONNSW.ENRGSRC* bit. When using the internal link-up signal, this bit should be set to 0b. 2. Should be set to 0b when using an internal copper PHY or when working in SGMII, 1000BASE-BX or 1000BASE-KX modes.
SPEED	9:8	10b	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. 00b = 10 Mb/s. 01b = 100 Mb/s. 10b = 1000 Mb/s. 11b = Not used.
Reserved	10	0b	Reserved. Write 0b, ignore on read.
FRCSPD	11	0b[1]	Force Speed This bit is set when software needs to manually configure the MAC speed settings according to the SPEED bits. Note that MAC and PHY must resolve to the same speed configuration or software must manually set the PHY to the same speed as the MAC. Software must clear this bit to enable the PHY or ASD function to control the MAC speed setting. Note that this bit is superseded by the *CTRL_EXT.SPD_BYPS* bit, which has a similar function.
FRCDPLX	12	0b	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, in 10/100/1000Base-T link mode, 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.
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.
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 Flash-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:***
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 Flash-configurable). If SDP1 is configured as an input, reads return the current value of the pin. ***Note:***

Field	Bit(s)	Initial Value	Description
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 Flash.
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 Flash-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 Flash-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.
RST (SC)	26	0b	Port Software Reset This bit performs a reset to the LAN port, resulting in a state nearly approximating the state following a power-up reset or internal PCIe reset, except for system PCI configuration and DMA logic. 0b = Normal. 1b = Reset. This bit is self clearing and is referred to as software reset or global reset.
RFCE	27	1b	Receive Flow Control Enable When set, indicates that the I210-CS/CL responds to the reception of flow control packets. If auto-negotiation is enabled, this bit is set to the negotiated flow control value. In SerDes mode the resolution is done by the hardware. In internal PHY, SGMII or 1000BASE-KX modes it should be done by the software.
TFCE	28	0b	Transmit Flow Control Enable When set, indicates that the I210-CS/CL 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. In SerDes mode the resolution is done by the hardware. In internal PHY, SGMII or 1000BASE-KX modes it should be done by the software.
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:*** 1. Asserting *DEV_RST* generates an interrupt via the ICR.DRSTA interrupt bit. 2. Device Reset (*CTRL.DEV_RST*) can be used to globally reset the entire component if the *DEV_RST_EN* bit in Initialization Control 4 Flash word is set. 3. Asserting *DEV_RST* sets the *STATUS.DEV_RST_SET* bit.
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 1000BASE-T PHY. 0b = Normal operation. 1b = Internal PHY reset asserted.

1. These bits are loaded from Flash.



7.2.2 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. 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. When the internal PHY is used, this reflects whether the PHY's LINK indication is present. When the SerDes, SGMII or 1000BASE-KX interface is used, this indicates loss-of-signal; if auto-negotiation is also enabled, this can also indicate successful auto-negotiation. Refer to Section 3.6.4 for more details. ***Note:*** This bit is valid only when working in internal PHY mode. In SerDes mode bit is always 0b.
Reserved	3:2	X	Reserved Write 0b, ignore on read.
Reserved	3:2	X	Reserved. Write 0, 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.
SPEED	7:6	X	Link Speed Setting Reflects the speed setting of the MAC and/or link when it is operating in 10/100/1000BASE-T mode (internal PHY). When the MAC is operating in 10/100/1000BASE-T mode with the internal PHY, these bits normally reflect the speed of the actual link, negotiated by the PHY and link partner and reflected internally from the PHY to the MAC (*SPD_IND*). These bits also might represent the speed configuration of the MAC only, if the MAC speed setting has been forced via software (*CTRL.SPEED*) or if MAC auto-speed detection is used. If auto-speed detection is enabled, the I210-CS/CL's speed is configured only once after the LINK signal is asserted by the PHY. 00b = 10 Mb/s. 01b = 100 Mb/s. 10b = 1000 Mb/s. 11b = 1000 Mb/s.
ASDV	9:8	X	Auto-Speed Detection Value Speed result sensed by the I210-CS/CL‚Äôs MAC auto-detection function. These bits are provided for diagnostics purposes only. The ASD calculation can be initiated by software writing a logic 1b to the *CTRL_EXT.ASDCHK* bit. The resultant speed detection is reflected in these bits. Refer to Section 7.2.3 for details.

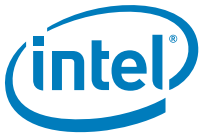
Field	Bit(s)	Initial Value	Description
PHYRA	10	1b	PHY Reset Asserted This read/write bit is set by hardware following the assertion of an internal PHY reset; it is cleared by writing a 0b to it. This bit is also used by firmware indicating a required initialization of the I210-CS/CL‚Äôs PHY.
Reserved	18:11	0x0	Reserved. Write 0b, ignore on read.
GIO Master Enable Status	19	1b	Cleared by the I210-CS/CL 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 (R/W1C)	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.
PF_RST_DONE	21	1b	PF _RST_DONE When set, indicates that software reset (*CTRL.RST*) or device reset (*CTRL.DEV_RST*) has completed and the software device driver can begin initialization process.
Reserved	30:22	0x0	Reserved. Write 0b, ignore on read.
MAC clock gating Enable	31	0b[1]	MAC Clock Gating Enable This bit is loaded from the Flash indicating that the device supports MAC clock. gating

1. If the signature bits of the Flash's Initialization Control Word 1 match (01b), this bit is read from the Flash.

7.2.3 Extended Device Control Register - CTRL_EXT (0x0018; R/W)

This register provides extended control of the I210-CS/CL'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.
I2C over SDP Enabled	1	0b[1]	Enable I^2C over SDP0 and SDP2 pins. When set, SDP0 and SDP2 pins functions as an I^2C interface operated through the I2CCMD,I2CPARAMS register set.
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.
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 Flash-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 Flash-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.



Field	Bit(s)	Initial Value	Description
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 Flash-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 Flash-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
ASDCHK	12	0b	Auto-Speed-Detection (ASD) Check Initiates an ASD sequence to sense the frequency of the PHY receive clock (RX_CLK). The results are reflected in *STATUS.ASDV*. This bit is self-clearing.
EE_RST (SC)	13	0b	EEPROM Block Reset When set, initiates a reset-like event to the EEPROM block function. This causes an Flash auto-load operation as if a software reset (*CTRL.RST*) had occurred. This bit is self-clearing.
Reserved	14	0x0	Reserved. Write 0b, ignore on read.
SPD_BYPS	15	0b	Speed Select Bypass When set to 1b, all speed detection mechanisms are bypassed, and the I210-CS/CL is immediately set to the speed indicated by *CTRL.SPEED*. This provides a method for software to have full control of the speed settings of the I210-CS/CL and when the change takes place, by overriding the hardware clock switching circuitry.
NS_DIS	16	0	No Snoop Disable When set to 1b, the I210-CS/CL 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, the I210-CS/CL 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, the I210-CS/CL requests relaxed ordering transactions as specified by registers RXCTL and TXCTL (per queue and per flow).
SerDes Low Power Enable	18	0b[1]	When set, enables the SerDes to enter a low power state when the function is in Dr state .
Dynamic MAC Clock Gating	19	0b[1]	When set, enables dynamic MAC clock gating.
Reserved	20	1b[1]	Reserved.
Reserved	21	0b	Reserved. Write 0b, ignore on read.
LINK_MODE	23:22	0x0[1]	Link Mode Controls interface on the link. 00b = Direct copper (1000Base-T) interface (10/100/1000 BASE-T internal PHY mode). 01b = 1000BASE-KX. 10b = SGMII. 11b = SerDes interface. ***Note:*** 1. This bit is reset only on power-up or PCIe reset.
Reserved	24	0b	Reserved. Write 0b, ignore on read.

Field	Bit(s)	Initial Value	Description
I2C Enabled	25	0b[1]	Enable I^2C This bit enables the SFPx_I2C pins that can be used to access external SFP modules or an external 1000BASE-T PHY via the MDIO interface. If cleared, the SFPx_I2C pads are isolated and accesses to the SFPx_I2C pins through the I2CCMD register or the MDIC register are ignored.
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 Flash auto load and should only be changed while Tx and Rx processes are stopped.
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. ***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 Flash.

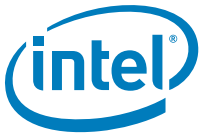
The I210-CS/CL enables up to four externally controlled interrupts. All software-definable pins, these can be mapped for use as GPI interrupt bits. Mappings are enabled by the *SDPx_GPIEN* bits only when these signals are also configured as inputs via *SDPx_IODIR*. When configured to function as external 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 7-8 for clarity.

Table 7-8. 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 Flash, which occurs after asserting the EE_RST bit.



Note: The Flash reset function can read configuration information out of the Flash, 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 the I210-CS/CL'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 the I210-CS/CL's clock switching circuitry performs the change.

7.2.4 Media Dependent Interface (MDI) Control Register - MDIC (0x0020; R/W)

Software uses this register to read or write MDI registers in the internal PHY or an external SGMII PHY.

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.
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 the I210-CS/CL 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 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.

7.2.5 MDC/MDIO Configuration Register – MDICNFG (0x0E04; R/W)

Note: This register is used to configure the MDIO connection that is accessed via the MDIC register. Refer to Section 3.6.2.1.2 for details on usage of this register.

Field	Bit(s)	Initial Value	Description
Reserved	20:0	0x0	Reserved. Write 0b, ignore on read.
PHYADD[1]	25:21	0x00	External PHY Address When the *MDICNFG.Destination* bit is 0b, default PHYADD accesses the internal PHY.
Reserved	30:26	0x0	Reserved. Write 0b, ignore on read.
Destination[2]	31	0b	Destination 0b = MDIO transactions using the MDIC register are directed to the internal PHY. 1b = MDIO transactions using the MDIC register are directed to an external PHY using the MDC/MDIO protocol. ***Note:*** When using the I2CCMD register to access an external PHY using the I^2C protocol, the *Destination* field must be set to 0b.

1. PHYADD is loaded from Initialization Control 4 Flash word to allocate the port address when using an external MDIO port.
2. Destination is loaded from Flash Initialization Control 3 word. When an external PHY supports a MDIO interface, this bit is set to 1b; otherwise, this bit is set to 0b.

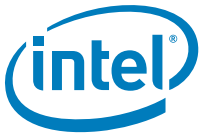
7.2.6 Fiber Switch Control - CONNSW (0x0034; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	1:0	00b	Reserved
ENRGSRC	2	0b[1]	SerDes Energy Detect Source 0b = SerDes Energy detect source is internal. 1b = SerDes Energy detect source is from SRDS_[n]_SIG_DET pin. This bit defines the source of the signal detect indication used to set link up while in SerDes mode. ***Note:*** In SGMII and 1000BASE-KX modes energy detect source is internal and value of *CONNSW.ENRGSRC* bit should be 0b.
Reserved	8:3	0x0	Reserved. Write 0x0, ignore on read.
SerDesD (RO)	9	X	SerDes Signal Detect Indication Indicates the SerDes signal detect value according to the selected source (either external or internal). Valid only if LINK_MODE is SerDes, 1000BASE-KX or SGMII.
Reserved	10	X	Reserved.
Reserved	11	X	Reserved.
Reserved	31:12	0x0	Reserved. Write 0x0, ignore on read.

1. The default value of the *ENRGSRC* bit in this register is defined in the Initialization Control 3 (Offset 0x24) Flash word (bit 15).

7.2.7 VLAN Ether Type - VET (0x0038; R/W)

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
VET EXT	31:16	0x8100	External VLAN Ether Type.

7.2.8 LED Control - LEDCTL (0x0E00; RW)

This register controls the setup of the LEDs. Refer to Section 6.5.1 for details of the *Mode* fields encoding.

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.
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). 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	13:12	0b	Reserved Write as 0x0,ignore on read.
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). 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.

Field	Bit(s)	Initial Value	Description
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). 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 Flash.

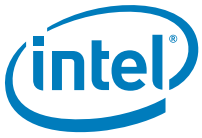
7.3 Internal Packet Buffer Size Registers

The following registers define the size of the on-chip receive and transmit buffers used to receive and transmit packets.

The registers in this section reset only on power up.

7.3.1 RX Packet Buffer Size - RXPBSIZE (0x2404; R/W)

Field	Bit(s)	Init.	Description
Rxpbsize	5:0	0x22	Rx packet buffer size in KB.
Bmc2ospbsize	11:6	0x02	BMC to OS packet buffer size in KB.
Reserved	30:12	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.



7.3.2 TX Packet Buffer Size - TXPBSIZE (0x3404; R/W)

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.

7.4 Flash Registers Descriptions

7.4.1 EEPROM-Mode Control Register - EEC (0x12010; RW)

This register provides software direct access to the Flash.
Bit banging access to the flash via the *FLA* register is not protected by this field.

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 Flash is present with a valid signature and the hardware was programmed from the Flash. The hardware will always first check the existence of the external Flash. 0b = Flash is not used. 1b = Flash is used.
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 either a Flash is present and has the correct signature field or the iNVM is no-empty, and the shadow RAM contains the auto-load information from one of those sources (no need for software programming).
Auto_RD (RO)	9	0b	Flash Auto-Read Done. When set to 1b, this bit indicates that the auto-read by hardware from the Flash is done. This bit is also set when the Flash is not found or when its signature field is not valid.
Reserved	10	0b	Reserved.
EE_Size (RO)	14:11	0101b	Flash 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 Kb units.
Reserved	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	22:20	0x0	Reserved. Reads as 0x0.
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.
Reserved	24	0b	Reserved. Reads as 0b.

Field	Bit(s)	Init.	Description
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.
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.
Reserved	31:27	0x0	Reserved. Reads as 0x0.

7.4.2 EEPROM-Mode Load Control/Status Register - EELOADCTL (0x12020; RO to Host, RW to FW)

This register provides software EEPROM-mode load status.

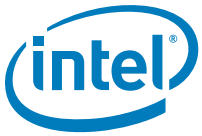
All bits are RW to FW, RO to host - excepted to bit 12.
Bit banging access to the flash via the *FLA* register is not protected by this field.

Field	Bit(s)	Init.	Description
ee-pcie-done_e (RO)	1	0	Indicates status of the NVM auto load section read following PCIe reset.
Reserved	2	1b	
Reserved	7:5	0b	Reserved.
Reserved	31:28	0x0	Reserved. Reads as 0b.

This register provides software direct access to the EEPROM. Software can control the EEPROM by successive writes to this register. Data and address information is clocked into the EEPROM by software toggling the *EE_SK* and *EE_DI* bits (0 and 2) of this register with *EE_CS* set to 0b. Data output from the EEPROM is latched into the *EE_DO* bit (bit 3) via the internal 62.5 MHz clock and can be accessed by software via reads of this register.

Bit banging access to the flash via the *FLA* register is not protected by this field.

Field	Bit(s)	Initial Value	Description
EE_SK	0	0b	Clock input to the EEPROM When *EE_GNT* = 1b, the *EE_SK* output signal is mapped to this bit and provides the serial clock input to the EEPROM. Software clocks the EEPROM via toggling this bit with successive writes.
EE_CS	1	1b	Chip select input to the EEPROM When *EE_GNT* = 1b, the *EE_CS* output signal is mapped to the chip select of the EEPROM device. Software enables the EEPROM by writing a 0b to this bit.
EE_DI	2	1b	Data input to the EEPROM When *EE_GNT* = 1b, the *EE_DI* output signal is mapped directly to this bit. Software provides data input to the EEPROM via writes to this bit.
EE_DO (RO)	3	X[1]	Data output bit from the EEPROM The *EE_DO* input signal is mapped directly to this bit in the register and contains the EEPROM data output. This bit is RO from a software perspective; writes to this bit have no effect.
FWE	5:4	01b	Flash Write Enable Control These two bits, control whether writes to Flash memory are allowed. 00b = Flash erase (along with bit 31 in the *FLA* register). 01b = Flash writes disabled. 10b = Flash writes enabled. 11b = Reserved.



Field	Bit(s)	Initial Value	Description
EE_REQ	6	0b	Request EEPROM Access The software must write a 1b to this bit to get direct EEPROM access. It has access when *EE_GNT* is 1b. When the software completes the access it must write a 0b.
EE_GNT	7	0b	Grant EEPROM Access When this bit is 1b the software can access the EEPROM using the SK, CS, DI, and DO bits.
EE_PRES (RO)	8	X	EEPROM Present and Signature is valid This bit indicates that an EEPROM is present and the value of the *Signature* field in the *EEPROM Sizing and Protected Fields* EEPROM word (Word 0x12) is 01b 0b = Signature field invalid 1b = EEPROM present and signature is valid.
Auto_RD (RO)	9	0b	EEPROM Auto Read Done When set to 1b, this bit indicates that the auto read by hardware from the EEPROM is done. This bit is also set when the EEPROM is not present or when its signature is not valid.
EE_ADDR_SIZE (RO)	10	1b	EEPROM Address Size This field defines the address size of the EEPROM. This bit is set by the EEPROM size auto-detect mechanism. If no EEPROM is present or the signature is not valid, a 16-bit address is assumed. 0b = 8- and 9-bit. 1b = 16-bit.
EE_SIZE (RO)	14:11[2]	0111b	EEPROM Size This field defines the size of the EEPROM: Field Value EEPROM Size EEPROM Address Size 0111b 16 Kbytes 2 bytes 1000b 32 Kbytes 2 bytes
EE_BLOCKED (RO)	15	0b	EEPROM access blocked EEPROM Bit Banging access blocked - Bit is set by HW when detecting an EEPROM access violation during bit banging access using the *EEC* register or detecting an EEPROM access violation when accessing the EPROM using the *EERD* register. When bit is set further Bit Banging operations from the function are disabled until bit is cleared. Type of violations that can cause the bit to be set are write to read-only sections, access to a hidden area or any other EEPROM protection violation detected. ***Note:*** Bit is cleared by write one to the *EEC.EE_CLR_ERR* bit.
EE_ABORT (RO)	16	0b	EEPROM access Aborted Bit is set by HW when EEPROM access was aborted due to deadlock avoidance, management reset or EEPROM reset via CTRL_EXT.EE_RST. When bit is set further Bit Banging operations from the Function are disabled until bit is cleared. ***Note:*** Bit is cleared by write one to the EEC.EE_CLR_ERR bit.
EE_RD_TIMEOUT (RO)	17	0b	EERD access timeout When bit is set to 1b indicates the EEPROM access via EERD register timed out while trying to read EEPROM status (Can occur when no EEPROM exists). ***Note:*** Bit is cleared by write one to the *EEC.EE_CLR_ERR* bit.

Field	Bit(s)	Initial Value	Description
EE_CLR_ERR (SC)	18	0b	Clear EEPROM Access Error A write 1b to the EE_CLR_ERR bit clears the *EEC.EE_ABORT* bit, *EE_BLOCKED* bit and *EE_RD_TIMEOUT* bit. ***Note:*** Clearing the *EEC.EE_ABORT* bit and *EE_BLOCKED* bit enables further Bit Banging access to the EEPROM from the function.
EE_DET (RO)	19	X	EEPROM Detected ***Note:*** Bit is set to 1b when EEPROM responded correctly to a get status opcode following power-up.
Reserved	31:20	0x0	Reserved Write 0 ignore on read.

1. Value depends on voltage level on EE_DO pin following initialization
2. These bits are read from the Flash.

7.4.3 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 Flash. To read a word, software writes the address to the *Read Address* field. Writing the register sets the Done bit to 0b. The I210-CS/CL 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.

This register is used by software to cause the I210-CS/CL to read individual words in the EEPROM. To read a word, software writes the address to the *Read Address* field and simultaneously writes a 1b to the *Start Read* field. The I211 reads the word from the EEPROM 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.

When this register is used to read a word from the EEPROM, that word does not influence any of the I210-CS/CL's internal registers even if it is normally part of the auto-read sequence.

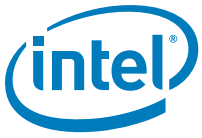
Note: Register reset on LAN_PWR_GOOD only.

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.
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.

7.4.4 EEPROM Load Error Register - EELOADERR (0x12028; RO)

This register indicates detection of errors in Hardware EEPROM load operation that are not CRC errors. Type of errors detected are:

— Reception of NACK when EEPROM attempts to write to a register.



— EEPROM read address exceeds EEPROM size.

— Length of section less than 2 in the *PCIe PHY Auto Configuration structure* and smaller than 3 in other CSR configuration structures.

— Length of section not an integer multiple of 2 in the *PCIe PHY Auto Configuration structure* and not an integer multiple of 3 in other CSR configuration structures.

Note: Register bits reset after LAN_PWR_GOOD or read without errors of appropriate EEPROM section.

7.4.5 EEPROM Load Control Register - EELOADCTL (0x12020; RW)

This register is used by software to control I211 auto-read operation and to execute EEPROM auto-read sequences mimicking occurrence of various resets.

Note: Register reset by LAN_PWR_GOOD only.

7.4.6 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 Flash. To write a word, software writes the address to the *Write Address* field and the data to the *Write Data* field. The I210-CS/CL 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 Flash, that word is not written to any of the I210-CS/CL's internal registers even if it is normally a hardware-accessed word.

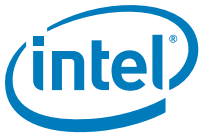
Field	Bit(s)	Init.	Description
CMDV (RO)	0	0b	Command Valid. This bit is cleared by hardware in case the write request was rejected.
DONE (RO field)	1	1b	Write Done. Set this bit to 1b when the EEPROM-mode write completes. Set this bit to 0b when the EEPROM-mode write is in progress. Note that writes 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.

7.4.7 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 Flash 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 Flash 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 Flash Secure mode.
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 Flash 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 Flash 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 Flash is in Secure mode. When set to 0b, the Flash is in Non-secure mode.
FLA_ABORT (RO)	7	0b[2]	Bit is 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	15:9	0b	Reserved. Reads as 0b.
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. 101b = 2 MB. 110b = 4 MB. 111b = 8 MB. This field is written by hardware from Flash words 0x11 after LAN_POWER_GOOD.
Reserved	28:20	0x0	Reserved.
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 Flash.
[2] These bits also reset when PCIe resets.



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 Flash by software toggling the *FL_SCK* bit (bit 0) of this register with *FL_CE* set to 0b. Data output from the Flash is latched into the *FL_SO* bit (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)	Initial Value	Description
FL_SCK	0	0b	Clock Input to the Flash When *FL_GNT* is 1b, the *FL_SCK* out signal is mapped to this bit and provides the serial clock input to the Flash device. Software clocks the Flash memory via toggling this bit with successive writes.
FL_CE	1	1b	Chip Select Input to the Flash When *FL_GNT* is 1b, the *FL_CE* output signal is mapped to the chip select of the Flash device. Software enables the Flash by writing a 0b to this bit.
FL_SI	2	1b	Data Input to the Flash When *FL_GNT* is 1b, the *FL_SI* output signal is mapped directly to this bit. Software provides data input to the Flash via writes to this bit.
FL_SO	3	X	Data Output Bit from the Flash The FL_SO input signal is mapped directly to this bit in the register and contains the Flash memory serial data output. This bit is read only from the software perspective – writes to this bit have no effect.
FL_REQ	4	0b	Request Flash Access The software must write a 1b to this bit to get direct Flash memory access. It has access when *FL_GNT* is 1b. When the software completes the access it must write a 0b.
FL_GNT	5	0b	Grant Flash Access When this bit is 1b, the software can access the Flash memory using the *FL_SCK*, *FL_CE*, *FL_SI*, and *FL_SO* bits.
FLA_add_size	6	0b	Flash Address Size 0b = Flash devices are accessed using 2 bytes of address. 1b = Flash devices (including 64 KB) are accessed using 3 bytes of the address. ***Notes:*** 1. If this bit is set by one of the functions, it is also reflected in all other functions. 2. If value of *BARCTRL.FLSize* field is greater than 0x0, bit is read as 1b.
FLA_ABORT (RO)	7	0b	Flash Access Aborted Bit is set by HW when Flash access was aborted due to deadlock avoidance. When bit is set further Flash Bit Banging access from the function are blocked. ***Note:*** Bit is cleared by write 1b to the FLA.FLA_CLR_ERR bit.
FLA_CLR_ERR (SC)	8	0b	Clear Flash Access Error A write 1b to the FLA_CLR_ER bit clears the FLA.FLA_ABORT bit and enables further Bit Banging access to the Flash from the function.
Reserved	28:9	0x0	Reserved Write 0 ignore on read.
FL_BAR_WR (RO)	29	0b	Flash Write via BAR in Progress This bit is set to 1b while a write to the Flash memory is in progress or is pending as a result of a direct Memory access (not bit banging access). When this bit is clear (read as 0b) software can initiate a byte write operation to the Flash device.
FL_BUSY (RO)	30	0b	Flash Busy When set to 1b indicates that a Flash memory access is in progress.
FL_ER (SC)	31	0b	Flash Erase Command When bit is set to 1b an erase command is sent to the Flash component only if the *EEC.FWE* field is 00b (Flash Erase). This bit is automatically cleared when flash erase has completed.

7.4.8 Flash Opcode - FLASHOP (0x12054; R/W)

This register enables the host or the firmware to define the op-code used in order to erase a sector of the flash or the complete flash. This register is reset only at power on or LAN_PWR_GOOD assertion.

This register is common to all ports. Register should be programmed according to the parameters of the flash used.

Note: The default values fit to Adesto* (formally Atmel*) Serial Flash Memory devices.

Notes:

1. Register reset on LAN_PWR_GOOD only.
2. Register shared by all functions.

Field	Bit(s)	Initial Value	Description
DERASE	7:0	0x0062	Flash Device Erase Instruction The op-code for the Flash erase instruction.
SERASE	15:8	0x0052	Flash Block Erase Instruction The op-code for the Flash block erase instruction.
Reserved	31:16	0x0	Reserved Write 0 ignore on read.

7.4.9 EEPROM Diagnostic - EEDIAG (0x1038; RO)

This register reflects the values of EEPROM bits influencing the hardware that are not reflected otherwise.

Note: Register shared by all functions.

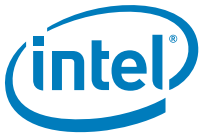
7.4.10 EEPROM Auto Read Bus Control - EEARBC (0x12024; R/W)

In EEPROM-less implementations, this register is used to program the I210-CS/CL the same way it should be programmed if an EEPROM was present.

This register is common to all functions and should be accessed only following access coordination with the other ports.

Notes:

1. Register reset on LAN_PWR_GOOD only.
2. Register shared by all functions.



Field	Bit(s)	Initial Value	Description
VALID_CORE0	0	0b	Valid Write Active to Core 0 Write strobe to Core 0. Firmware/software sets this bit for write access to registers loaded from EEPROM words in LAN0 section. Software should clear this bit to terminate the write transaction.
VALID_CORE1	1	0b	Valid Write Active to Core 1 Write strobe to Core 1. Firmware/software sets this bit for write access to registers loaded from EEPROM words in LAN1 section. Software should clear this bit to terminate the write transaction.
VALID_COMMON	2	0b	Valid Write Active to Common Write strobe to Common. Firmware/software sets this bit for write access to registers loaded from EEPROM words that are common to all sections. Software should clear this bit to terminate the write transaction.
VALID_PCIE	3	0b	Valid Write Active to PCIe PHY Write strobe to PCI PHY. Firmware/software sets this bit for write access to registers loaded from EEPROM words pointed by word 0x10 that are directed to the PCIe PHY. Software should clear this bit to terminate the write transaction.
ADDR	12:4	0x0	Write Address This field specifies the address offset of the EEPROM word from the start of the EEPROM Section. Sections supported are: • Common and LAN0 • LAN1 • LAN2 • LAN3
VALID_CORE2	13	0b	Valid Write Active to Core 2 Write strobe to Core 2. Firmware/software sets this bit for write access to registers loaded from EEPROM words in LAN2 section. Software should clear this bit to terminate the write transaction.
VALID_CORE3	14	0b	Valid Write Active to Core 3 Write strobe to Core 3. Firmware/software sets this bit for write access to registers loaded from EEPROM words in LAN3 section. Software should clear this bit to terminate the write transaction.
Reserved	15	0b	Reserved Write 0, ignore on read.
DATA	31:16	0x0	Data written into the EEPROM auto read bus.

1. Not all EEPROM addresses are part of the auto read. By using this register software can write to the hardware registers that are configured during auto read only.
2. Host access via *EEARBC* can be done only when no EEPROM presence is detected. Management can access the internal registers via *EEARBC* also when EEPROM presence is detected and EEPROM load is done.

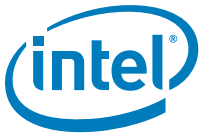
7.4.11 Management-EEPROM CSR I/F

The following registers are reserved for Firmware access to the EEPROM and are not writable by the host.

7.4.11.1 Management EEPROM Control Register - EEMNGCTL (0x12030; RW)

Field	Bit(s)	Initial Value	Description
ADDR	14:0	0x0	Address - This field is written along with Start Read or Start write to indicate the EEPROM word address to read or write.
START	15	0b	Start - Writing a 1b to this bit causes the EEPROM to start the read or write operation according to the write bit. ***Note:*** Bit is not cleared by Firmware reset.
WRITE	16	0b	Write - This bit tells the EEPROM if the current operation is read or write: 0b = read 1b = write
EEBUSY	17	0b	EEPROM Busy - This bit indicates that the Flash is busy processing an Flash transaction and Flash access will be delayed.
CFG_DONE 0[1]	18	0b	Configuration cycle is done for port 0 – This bit indicates that configuration cycle (configuration of SerDes, PHY, PCIe and PLLs) is done for port 0. This bit is set to 1b to indicate configuration done, and cleared by hardware on any of the reset sources that causes initialization of the PHY. Note: Port 0 driver should not try to access the PHY for configuration before this bit is set.
CFG_DONE 1[1]	19	0b	Configuration cycle is done for port 1 – This bit indicates that configuration cycle (configuration of SerDes, , PCIe and PLLs) is done for port 1. This bit is set to 1b to indicate configuration done, and cleared by hardware on any of the reset sources that cause initialization of the PHY. Note: Port 1 driver should not try to access the PHY for configuration before this bit is set.
CFG_DONE 2[1]	20	0b	Configuration cycle is done for port 2 – This bit indicates that the configuration cycle (configuration of SerDes, PCIe and PLLs) is done for port 2. This bit is set to 1b to indicate configuration done, and cleared by hardware on any of the reset sources that cause initialization of the PHY. Note: Port 2 driver should not try to access the PHY for configuration before this bit is set.
CFG_DONE 3[1]	21	0b	Configuration cycle is done for port 3 – This bit indicates that the configuration cycle (configuration of SerDes, PCIe and PLLs) is done for port 3. This bit is set to 1b to indicate configuration done, and cleared by hardware on any of the reset sources that cause initialization of the PHY. Note: Port 3 driver should not try to access the PHY for configuration before this bit is set.
Reserved	28:22	0x0	Reserved Write 0, ignore on read.
EEMNGCTL_CLR_ERR (SC)	29	0b	Clear Timeout Error A write 1b to the *EEMNGCTL.EEMNGCTL_CLR_ERR* bit clears the error reported in the *EEMNGCTL.TIMEOUT* bit.
TIMEOUT	30	0b	When bit is set to 1b indicates that a transaction timed out while trying to read the Flash status (Occurs when no Flash exists). ***Notes:*** 1. To clear the bit Firmware should write 1b to the *EEMNGCTL.EEMNGCTL_CLR_ERR* bit. 2. Bit is not cleared by Firmware reset.
DONE	31	1b	Transaction Done - This bit is cleared after the Start Write or Start Read bit is set by the MNG and is set back again when the Flash write or read transaction is done. ***Note:*** Bit is not cleared by Firmware reset.

1. Bit relates to physical port. If LAN Function Swap (*FACTPS.LAN Function Sel* = 1) is done, Software should poll *CFG_DONE* bit of original port to detect end of PHY configuration operation.



7.4.11.2 Management EEPROM Read/Write data - EEMNGDATA (0x12034; RW)

Field	Bit(s)	Initial Value	Description
WRDATA	15:0	0x0	Write Data Data to be written to the Flash.
RDDATA (RO)	31:16	‚Äì	Read Data Data returned from the Flash read.

7.4.12 FLASH Arbitration Control and Debug – ARBDBG (0x12044; RW)

This register enables FLASH arbitration control if following a read or write operation the flash chip selected remains low. It provides separate controls for each client of the FLASH.

This register also enables viewing which client currently is granted access to the flash.

7.4.13 EEPROM-Mode Auto Read Bus Control - EEARBC (0x12024; RO in Secured Mode)

Note: In iNVM implementations, this register is used to program the I210-CS/CL the same way it should be programmed if an NVM was present. This register is reset on LAN_PWR_GOOD only.

Field	Bit(s)	Initial Value	Description
VALID	0	0b	Valid. Indicates that the last auto-load bus write request is valid.
DONE	1	1b	Done. Last auto-load bus write request completed. The register can be written again with a new auto-load write request.
Reserved	3:2	0x0	Reserved. Write 0x0, ignore on read.
ADDR	12:4	0x00	Write Address. This field specifies the address offset of the Flash word from the start of the shadow RAM section[1].
Reserved	15:13	0x0	Reserved. Write 0x0, ignore on read.
DATA	31:16	0x0000	Data written into the Flash auto read bus.

1. Not all shadow RAM addresses are part of the auto read (auto-load). By using this register software can write to the hardware registers that are configured during auto load only.

7.4.14 Flash Mode Register – FLASHMODE (0x12000; RO in Secured Mode)

This register controls the interface for the Flash device. This register is reset only at power on or during LAN_PWR_GOOD assertion. It is loaded by firmware from its own module.

Field	Bit(s)	Init.	Description
FAST_READ_MODE	0	0b	When set to 1b, the op-code for a Read command is taken from FLASHOP.FASTREAD and the number of dummy bytes are asserted as indicated in the NUM_OF_DUMMY field.

Field	Bit(s)	Init.	Description
NUM_OF_DUMMY	2:1	01b	Indicates the number of dummy bytes that should be provided to the Flash after providing the address.
FLASH_SPEED	4:3	0b	Indicates the frequency of the clock provided to the Flash. 00b = Clock is 15.125 MHz 01b = Clock is 31.25 MHz. 10b = Clock is 62.5 MHz. 11b = Reserved.
Reserved	5	0	Reserved
SST_MODE	6	1b	When set to 1b,indicates that the current Flash device operates in a 1-byte program for each Flash access. This mode is the default operating mode as it is supported by all Flash devices. However, each time the Flash device supports burst writes, clearing this bit improves Flash write performance.
Reserved	31:7	0x0	Reserved.

7.4.15 Flash Op-code Register – FLASHOP (0x12054; RO in Secured Mode)

This register holds the Flash op-codes. This register is reset only at power on or during LAN_PWR_GOOD assertion. It is loaded by firmware from its own module.

Field	Bit(s)	Init.	Description
FLASHERASEOP	7:0	0xC7	Holds the op-code for erasing the entire Flash device.
SUSPENDOP	15:8	0x75	Holds the op-code for suspending the program/erase operation in the Flash.
RESUMEOP	23:16	0x7A	Holds the op-code for resuming the program/erase operation that was suspended.
FASTREADOP	31:24	0x0B	Holds the op-code that is issued in a Read command when Fast Read Mode Is set.

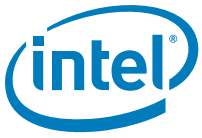
7.4.16 FLASH General Purpose OP-Code Register – FLASHGOP (0x12058; RO in secured mode)

This register holds the Flash Read/write general purpose op-codes. This register is reset only at power on or during LAN_PWR_GOOD assertion. It is loaded by firmware from its own module.

7.4.17 Flash Timing Register – FLASHTIME (0x12004; RO in Secured Mode)

This register holds the timing parameters for Flash access. This register is reset only at power on or during LAN_PWR_GOOD assertion. It is loaded by firmware from its own module.

Field	Bit(s)	Init.	Description
CSDESELECT	3:0	0xB	Indicates the time in cycles of 8 ns that CS should be de-asserted between two commands. Note that an offset of 16 ns is added to the programmed value. The default is 104 ns.
Reserved	15:04	0x0	Reserved.
HOLDTIME	31:16	0x00FF	The I210-CS/CL maintains a hold timer that counts the time that CS is asserted and no command is issued. When the timer expires, the CS is de-asserted and the next command starts a new transaction to Flash. The time is measured in cycles of 16 ns. The default is 4 µs.



7.4.18 FLASH Read Status Register – FLASHRDST (0x12008; RW)

This register holds the last read status from Flash. This register is reset only at power on or during LAN_PWR_GOOD assertion.

7.4.19 Flash Block Base Address – FLBLKBASE (0x12100; RO)

Field	Bit(s)	Init.	Description
Start Address	11:0	0x000	The base address expressed in a 4 KB sector index of the Flash section, which is protected from software writes. Aligned to 4 KB boundaries. Loaded from the secured section in Flash (word 0x10).
Reserved	30:12		Reserved.

7.4.20 Flash Block End Address – FLBLKEND (0x12104; RO)

Field	Bit(s)	Init.	Description
End Address	11:0	0x000	The last 4 KB sector index included in the Flash section, which is protected from software writes. It is derived by firmware from Max Module Area field extracted from the new firmware image header. A null value in this field means no blocked area.
Reserved	31:12		Reserved

7.4.21 Flash Firmware Code Update – FLFWUPDATE (0x12108; RW)

Field	Bit(s)	Init.	Description
Reserved	28:0		Reserved.
AUTHEN-DONE (RO)	29	0x0	Authentication Cycle Done. Set to 1b when done. This bit is self-cleared once the update request is set to 1b.
AUT_FAIL (RO)	30	0x0	Authentication failed. Set to 1b 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.

7.4.22 Shadow RAM Debug – SHADOWDBG (0x1206C; RW)

7.4.23 EEPROM-Mode Diagnostic - EEDIAG (0x12060; RO)

This register reflects the values of NVM bits influencing the hardware that are not reflected otherwise.

7.4.24 EEPROM Block Base Address – EEBLKBASE (0x1210C; RO)

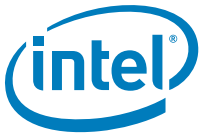
Field	Bit(s)	Init.	Description
1st Start Address	10:0	0x000	The base address expressed in words of the first hardware section (EEPROM map), which is protected from software writes. Loaded from the secured section in the Flash (word 0x2D).
Reserved	11		Reserved.
2nd Start Address	22:12	0x000	The base address expressed in words of the second hardware section (EEPROM map), which is protected from software writes. Loaded from the secured section in Flash (word 0x12). This read-only section ends at the shadow RAM ends. For legacy reasons, it is cleared to 0x000 when there is no second hardware protected section in the shadow RAM.
Reserved	31:23		Reserved.

7.4.25 EEPROM Block End Address – EEBLKEND (0x12110; RO)

Field	Bit(s)	Init.	Description
1st End Address	10:0	0x000	The last address expressed in words of the first hardware section (EEPROM map), which is protected from software writes. Loaded from the secured section in the Flash (word 0x2C). For legacy reasons, it is cleared to 0x000 when there is no first hardware protected section in the shadow RAM.
Reserved	31:11		Reserved

7.4.26 Software Flash Burst Control Register - FLSWCTL (0x12048; RW)

Field	Bit(s)	Init.	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	00b	Command. Indicates which command that should be executed.
CMDV (RO)	28	0b	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 (RO)	29	0b	Flash Busy. This bit indicates that the Flash is busy processing a Flash transaction and should not be accessed.
DONE (RO)	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 (RO)	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}	FLSWCNT.CNT range	Limitations to Host	Command Description
0000b	1 B - 4 KB		Read
0001b	1 B - 256 B the write must not cross a page (256 B) boundary	When in the Flash Secure mode, this command is operational only if applied on un-secured words.	Write
0010b	Don't Care	When in Flash secure mode, this command is operational only if applied on un-secured sectors.	Flash sector (4 KB) erase (when no security). The 4 KB sector index to be erased is determined by the ADDR field.
0011b	Don't Care	This command is not operational when in Flash secure mode.	Flash device erase (when no security). The entire Flash device is erased.
0100b	1 B - 4 B		Read Status register of Flash device.
0101b	1 B - 4 B	This command is not operational when in Flash secure mode.	Write Status register of Flash device.
0110b	Don't Care	This command is not operational when in Flash secure mode.	Write Enable. Depending on the Flash device Datasheet, this command might be needed prior to issuing the 1100b Programmable Write Status register op-code.
0110b, 0111b			Reserved.
1000b	1 B - 4 B		Read JEDEC ID.
1001b			Reserved.
1010b			Reserved.
1011b	1 B - 4 B	Op-code cannot be re-programmed when in Flash secured mode	Programmable Read Status register (op-code 0x35 by default that can be re-programmed in FLASHGOP).
1100	0 B - 4 B	This command is not operational when in Flash secure mode.	Programmable Write Status register (op-code 0x31 by default that can be re-programmed in FLASHGOP).

7.4.27 Software Flash Burst Data Register - FLSWDATA (0x1204C; RW)

Field	Bit(s)	Init.	Description
DATA	31:0	0x0	Burst Flash Data. Data written to or read from the Flash. 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.

7.4.28 Software Flash Burst Access Counter - FLSWCNT (0x12050; RW)

Field	Bit(s)	Init.	Description
Reserved	31:13	0x0	Reserved
CNT	12:0	0x0	Flash Burst Counter. This counter holds the size in bytes of the Flash burst read or write.

7.4.29 Data - INVM_DATA (0x12120 + 4*n [n = 0..63]; R/W1)

These registers holds the iNVM memory content. The iNVM memory is organized in 32 lines of 64-bits each. INVM_DATA[0] holds the lowest 32 bits of the first iNVM line. INVM_DATA[1] holds the highest 32 bits of the first iNVM line.

Field	Bit(s)	Init.	Description
DATA	31:0	0x0	Data value programmed or to be programmed in the corresponding iNVM line segment (high or low order 32-bits). Once a bit that has been programmed to 1b, it cannot be re-programmed to 0b.

7.4.30 Lock - INVM_LOCK (0x12220 + 4*n [n = 0..31]; R/W1)

The iNVM memory is organized in 32 lines of 64-bits each. INVM_LOCK[n] controls iNVM memory line n.

Field	Bit(s)	Init.	Description
LOCK	0	0b	When set to 1b, the corresponding iNVM line is locked and cannot be programmed. Once this bit that has been programmed to 1b, it cannot be re-programmed to 0b.
Reserved	31:1	0x0	Reserved.

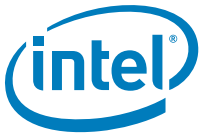
7.4.31 Test - INVM_TEST (0x122A0 + 4*n [n = 0..31]; R/W1)

The iNVM memory is organized in 32 lines of 64-bits each. INVM_TEST[n] is relative to memory line n. The test bits are reserved for manufacturing tests. The chip may arrive with any value in these bits. They have no impact on the chip behavior. User may use any test bit that was not already set to test his iNVM write function.

7.4.32 Protect - INVM_PROTECT (0x12324; RW)

Note: Register bits reset after LAN_PWR_GOOD.

Field	Bit(s)	Init.	Description
ALLOW_WRITE (RO)	0	0b	When read as 1b, it indicates that the iNVM is enabled for writes.For example, the *Code* field was written with the correct value, which enables writing the iNVM (0xABACADA).
WRITE_ERROR (RO)	1	0b	When read as 1b, it indicates an attempt to write the iNVM when the write was locked (ALLOW_WRITE bit was 0b) or when the iNVM was still busy with a previous write. This bit is cleared on the next iNVM write operation that was successfully performed.
BUSY (RO)	2	0b	When read as 1b, it indicates that the iNVM is still busy with the previous write.
Reserved	3	0b	Reserved.
CODE	31:4	0x0	When set to 0xABACADA, the iNVM is enabled for writes. Any other value set in this field is protected against mistakenly writing the iNVM. This field is always read as 0x0.



7.5 Flow Control Register Descriptions

7.5.1 Flow Control Address Low - FCAL (0x0028; RO)

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 the I210-CS/CL. Whether flow control packets are passed to the host (software) depends on the state of the *RCTL.DPF* bit.

Field	Bit(s)	Initial Value	Description
FCAL	31:0	0x00C28001	Flow Control Address Low.

7.5.2 Flow Control Address High - FCAH (0x002C; RO)

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.

Field	Bit(s)	Initial Value	Description
FCAH	15:0	0x0100	Flow Control Address High. Should be programmed with 0x01_00.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

7.5.3 Flow Control Type - FCT (0x0030; R/W)

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].

Field	Bit(s)	Initial Value	Description
FCT	15:0	0x8808	Flow Control Type.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

7.5.4 Flow Control Transmit Timer Value - FCTTV (0x0170; R/W)

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.

7.5.5 Flow Control Receive Threshold Low - FCRTL0 (0x2160; R/W)

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).

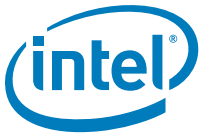
Flow control reception/transmission are negotiated capabilities by the auto-negotiation process. When the I210-CS/CL is manually configured, flow control operation is determined by the *CTRL.RFCE* and *CTRL.TFCE* bits.

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0x0	Reserved. Write 0x0, ignore on read.
RTL	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.
Reserved	30:17	0x0	Reserved. Write 0x0, ignore on read.
XONE	31	0b	XON Enable. 0b = Disabled. 1b = Enabled.

7.5.6 Flow Control Receive Threshold High - FCRTH0 (0x2168; R/W)

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 the I210-CS/CL 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.
RTH	17:4	0x0	Receive Threshold High. FIFO 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 3.6.5.3.1 for calculation of *FCRTH0.RTH* value. ***Notes:*** 1. When in DMA coalescing operation and the internal transmit buffer is empty, the threshold high value defined in *FCRTC.RTH_Coal* is used instead of the *FCRTH0.RTH* value to allow an increase of the receive threshold high value by the maximum supported Jumbo frame size. 2. The value programmed should be greater than the maximum packet size.
Reserved	31:18	0x0	Reserved. Write 0x0, ignore on read.

7.5.7 Flow Control Refresh Threshold Value - FCRTV (0x2460; R/W)

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.

7.5.8 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.

7.6 PCIe Register Descriptions

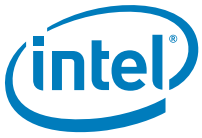
7.6.1 PCIe Control - GCR (0x5B00; RW)

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 Flash.
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	30:19	0x0	Reserved.
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* Flash word (word 0x15).
2. The default value for this field is read from the *PCIe Init Configuration 3* Flash 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.

7.6.2 PCIe Statistics Control #1 - GSCL_1 (0x5B10; RW)

Field	Bit(s)	Initial Value	Description
GIO_COUNT_EN_0	0	0b	Enable PCIe Statistic Counter Number 0.
GIO_COUNT_EN_1	1	0b	Enable PCIe Statistic Counter Number 1.
GIO_COUNT_EN_2	2	0b	Enable PCIe Statistic Counter Number 2.
GIO_COUNT_EN_3	3	0b	Enable PCIe Statistic Counter Number 3.
LBC Enable 0	4	0b	When set, statistics counter 0 operates in Leaky Bucket mode.
LBC Enable 1	5	0b	When set, statistics counter 1 operates in Leaky Bucket mode.
LBC Enable 2	6	0b	When set, statistics counter 2 operates in Leaky Bucket mode.
LBC Enable 3	7	0b	When set, statistics counter 3 operates in Leaky Bucket mode.
Reserved	26:8	0x0	Reserved. Write 0x0, ignore on read.
GIO_COUNT_TEST	27	0b	Test Bit. Forward counters for testability.



Field	Bit(s)	Initial Value	Description
GIO_64_BIT_EN	28	0b	Enable two 64-bit counters instead of four 32-bit counters.
GIO_COUNT_RESET	29	0b	Reset indication of PCIe statistical counters.
GIO_COUNT_STOP	30	0b	Stop indication of PCIe statistical counters.
GIO_COUNT_START	31	0b	Start indication of PCIe statistical counters.

7.6.3 PCIe Statistics Control #2 - GSCL_2 (0x5B14; RW)

This register configures the events counted by the GSCN_0, GSCN_1, GSCN_2 and GSCN_3 counters.

Field	Bit(s)	Initial Value	Description
GIO_EVENT_NUM_0	7:0	0x0	Event type that counter 0 (GSCN_0) counts.
GIO_EVENT_NUM_1	15:8	0x0	Event type that counter 1 (GSCN_1) counts.
GIO_EVENT_NUM_2	23:16	0x0	Event type that counter 2 (GSCN_2) counts.
GIO_EVENT_NUM_3	31:24	0x0	Event type that counter 3 (GSCN_3) counts.

Table 7-9 lists the encoding of possible event types counted by GSCN_0, GSCN_1, GSCN_2 and GSCN_3.

Table 7-9. PCIe Statistic Events Encoding

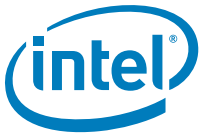
Transaction Layer Events	Event Mapping (Hex)	Description
Bad TLP From LL	0x0	For each cycle, the counter increases by one, if a bad TLP is received (bad CRC, error reported by AL, misplaced special char, or reset in thI of received tlp).
Requests That Reached Timeout	0x10	Number of requests that reached time out.
NACK DLLP Received	0x20	For each cycle, the counter increases by one, if a message was transmitted.
Replay Happened in Retry-Buffer	0x21	Occurs when a replay happened due to a timeout (not asserted when replay initiated due to NACK.
Receive Error	0x22	Set when one of the following occurs: 1. Decoder error occurred during training in the PHY. It is reported only when training ends. 2. Decoder error occurred during link-up or until the end of the current packet (if the link failed). This error is masked when entering/exiting EI.
Replay Roll-Over	0x23	Occurs when a replay was initiated for more than three times (threshold is configurable by the PHY CSRs).
Re-Sending Packets	0x24	Occurs when a TLP is resent in case of a completion timeout.
Surprise Link Down	0x25	Occurs when link is unpredictably down (not because of reset or DFT).
LTSSM in L0s in both Rx & Tx	0x30	Occurs when LTSSM enters L0s state in both Tx and Rx.
LTSSM in L0s in Rx	0x31	Occurs when LTSSM enters L0s state in Rx.
LTSSM in L0s in Tx	0x32	Occurs when LTSSM enters L0s state in Tx.
LTSSM in L1 active	0x33	Occurs when LTSSM enters L1-active state (requested from host side). ***Note:*** In case of RECOVERY entries not due to L1 exit, if the host will NAK the L1 request, there will be false L1 entry counts.
LTSSM in L1 SW	0x34	Occurs when LTSSM enters L1-switch (requested from switch side). ***Note:*** In case of RECOVERY entries not due to L1 exit, if the host will NAK the L1 request, there will be false L1 entry counts.
LTSSM in Recovery	0x35	Occurs when LTSSM enters recovery state.

7.6.4 PCIe Statistic Control Register #5...#8 - GSCL_5_8 (0x5B90 + 4*n[n=0...3]; RW)

These registers control the operation of the statistical counters GSCN_0, GSCN_1, GSCN_2 and GSCN_3 when operating in Leaky Bucket mode:

— GSCL_5 controls operation of GSCN_0.
— GSCL_6 controls operation of GSCN_1.
— GSCL_7 controls operation of GSCN_2.
— GSCL_8 controls operation of GSCN_3.

Field	Bit(s)	Initial Value	Description
LBC threshold n	15:0	0x0	Threshold for the Leaky Bucket Counter n.
LBC timer n	31:16	0x0	Time period between decrementing the value in Leaky Bucket Counter n.



7.6.5 PCIe Counter #0 - GSCN_0 (0x5B20; RC)

Field	Bit(s)	Initial Value	Description
EVC	31:0	0x0	Event Counter. Type of event counted is defined by the *GSCL_2.GIO_EVENT_NUM_0* field. Count value does not wrap around and remains stuck at the maximum value of 0xFF...F. Value is cleared by read.

7.6.6 PCIe Counter #1 - GSCN_1 (0x5B24; RC)

Field	Bit(s)	Initial Value	Description
EVC	31:0	0x0	Event Counter. Type of event counted is defined by the *GSCL_2.GIO_EVENT_NUM_1* field. Count value does not wrap around and remains stuck at the maximum value of 0xFF...F. Value is cleared by read.

7.6.7 PCIe Counter #2 - GSCN_2 (0x5B28; RC)

Field	Bit(s)	Initial Value	Description
EVC	31:0	0x0	Event Counter. Type of event counted is defined by the *GSCL_2.GIO_EVENT_NUM_2* field. Count value does not wrap around and remains stuck at the maximum value of 0xFF...F. Value is cleared by read.

7.6.8 PCIe Counter #3 - GSCN_3 (0x5B2C; RC)

Field	Bit(s)	Initial Value	Description
EVC	31:0	0x0	Event Counter. Type of event counted is defined *by the* GSCL_2.GIO_EVENT_NUM_3 field. Count value does not wrap around and remains stuck at the maximum value of 0xFF...F. Value is cleared by read.

7.6.9 Mirrored Revision ID - MREVID (0x5B64; R/W)

Field	Bit(s)	Initial Value	Description
Flash RevID	7:0	0x0	Mirroring of revision ID loaded from the Flash in PCIe configuration space (from Device Rev ID word, address 0x1E).
Step REV ID	15:8	0x01 for A1 0x03 for A2	Revision ID from FUNC configuration space.
Reserved	31:16	0x0	Reserved Write 0x0, ignore on read.

7.6.10 PCIe Control Extended Register - GCR_EXT (0x5B6C; RW)

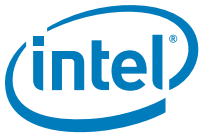
Field	Bit(s)	Init.	Description
Reserved	3:0	0x0	Reserved.
APBACD	4	0b	Auto PBA Clear Disable. When set to 1b, software can clear the PBA only by a direct write to clear access to the PBA bit. When set to 0b, any active PBA entry is cleared on the falling edge of the appropriate interrupt request to the 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	31:5	0x0	Reserved.

7.6.11 PCIe BAR Control - BARCTRL (0x5BFC; R/W) Target

Field	Bit(s)	Initial Value	Description
Reserved	31:0	0x0	Reserved Write 0x0, ignore on read.

Table 7-10. Usable Flash Size and CSR Mapping Window Size

FLBARSize	CSRSize	Resulted CSR + Flash BAR Size	Installed Flash Device	Usable Flash Space
000b	0	128 KB	No Flash	0
000b	1	256 KB	64 KB	64 KB
001b	0	256 KB	128 KB	128 KB
001b	1	n/a	n/a	Reserved
010b	0	256 KB	256 KB	256 KB minus 128 KB
010b	1	512 KB	256 KB	256 KB
011b	0	512 KB	512 KB	512 KB minus 128 KB
011b	1	1 MB	512 KB	512 KB
100b	0	1 MB	1 MB	1 MB minus 128 KB
100b	1	2 MB	1 MB	1 MB
101b	0	2 MB	2 MB	2 MB minus 128 KB
101b	1	4 MB	2 MB	2 MB
110b	0	4 MB	4 MB	4 MB minus 128 KB
110b	1	8 MB	4MB	4 MB
111b	0	8 MB	8MB	8 MB minus 128 KB
111b	1	16 MB	8MB	8 MB



7.6.12 Read Request To Data Completion Delay Register - RR2DCDELAY (0x5BF4; RC)

Note: Register resets by LAN_PWR_GOOD and PCIe reset.

Field	Bit(s)	Initial Value	Description
Max Split Time	31:0	0x0	This field captures the maximum PCIe split time in 16 ns units, which is the maximum delay between the read request to the first data completion. This is giving an estimation of the PCIe round trip time.

7.7 Semaphore Registers

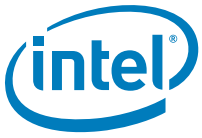
This section contains registers used to coordinate between firmware and software..

7.7.1 Software Semaphore - SWSM (0x5B50; R/W)

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 I210-CS/CL 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.
Reserved	31	0b	Reserved.

7.7.2 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 = Default mode for all SKUs. 001b = The I210-CS/CL mode. A proxy code was loaded. 010b = PT mode. I210-CS/CL SKUs only. 011b = Reserved. 100b = Host interface only. In the I210-CS/CL, this bit determines that a valid firmware code is running from the Flash but PT mode is disabled.
Reserved	5:4	00b	Reserved. Write 0x0, ignore on read.
EEP_Reload_Ind	6	0b	Flash Reloaded Indication. Set to 1b after firmware reloads the Firmware related sections of the Flash. Cleared by firmware at Firmware reset only.
Reserved	14:7	0x0	Reserved Write 0x0, ignore on read.
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 the I210-CS/CL, firmware should set this bit to 1b when it is ready (end of boot sequence).
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.



Field[1]	Bit(s)	Initial Value	Description
Ext_Err_Ind	24:19	0x0	External Error Indication Firmware writes here the reason that the firmware operation has stopped. For example, Flash CRC error, etc. Possible values: 0x00: No Error. 0x01: Flash CRC error in test configuration module. Reserved. 0x03: Flash CRC error in common firmware parameters module. 0x04: Flash CRC error in pass through. 0x05: Shadow RAM dump fault. 0x06: Bad Flash contents. 0x07: Reserved. 0x08: Flash CRC error in sideband configuration module. 0x09: Flash CRC error in flexible TCO filter configuration module. 0x0A: Flash CRC Error in NC-SI microcode download module. 0x0B: Flash CRC Error in NC-SI configuration module. 0x0C: Flash CRC Error in traffic type parameters module. 0x0D: Flash CRC Error in inventory Flash structure module. 0x0E: Flash 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: Flash CRC Error in hardware auto-load. 0x21: No Flash. 0x22: TCO isolate mode active. 0x23: Management memory parity error. 0x24: Firmware Flash access failure. 0x25: Other management error detected. 0x26 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.
PHY_SERDES0_Config_ Err_Ind	26	0b	PHY/SerDes Configuration Error Indication. Set to 1b by firmware when it fails to configure LAN PHY/SerDes. Cleared by firmware after successfully configuring LAN PHY/SerDes.
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. The software device driver should only read this register.
2. Firmware ignores the Flash semaphore in operating system hung states.
3. Bits 15:0 are cleared on firmware reset.

7.7.3 Software–Firmware Synchronization - SW_FW_SYNC (0x5B5C; RWM)

This register is intended to synchronize between software and firmware.

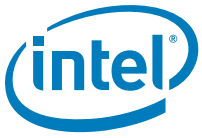
Note: If software takes ownership of bits in the *SW_FW_SYNC* register for a duration longer than 1 second, firmware can take ownership of the bit.

Field	Bit(s)	Initial Value	Description
SW_FLASH_SM	0	0b	When set to 1b, Flash access is owned by software.
SW_PHY_SM	1	0b	When set to 1b, SerDes/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.
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, Flash 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.
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 global 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.



7.8 Interrupt Register Descriptions

7.8.1 PCIe Interrupt Cause - PICAUSE (0x5B88; RW1/C)

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.RST*) reset to enable PCIe activity.
Reserved	31:6	0x0	Reserved. Write 0x0, ignore on read.

7.8.2 PCIe Interrupt Enable - PIENA (0x5B8C; R/W)

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.

7.8.3 Extended Interrupt Cause - EICR (0x1580; RC/W1C)

This register contains the frequent interrupt conditions for the I210-CS/CL. 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 7-11. 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 7-12. 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.

7.8.4 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 7-13. 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.

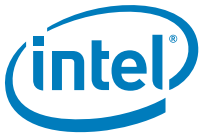


Table 7-14. EICS Register - MSI-X Mode (GPIE.Multiple_MSIX = 1b)

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.

7.8.5 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 7-15. 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 7-16. 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.

7.8.6 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 7-17. 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 7-18. 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.

7.8.7 Extended Interrupt Auto Clear - EIAC (0x152C; R/W)

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.

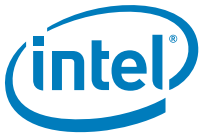
The MSI-X message can be delayed by *EITR* moderation from the time the *EICR* bit is activated.

Table 7-19. 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.

7.8.8 Extended Interrupt Auto Mask Enable - EIAM (0x1530; R/W)

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 7-20. 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 7-21. 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.

7.8.9 Interrupt Cause Read Register - ICR (0x1500; RC/W1C)

This register contains the interrupt conditions for the I210-CS/CL 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.

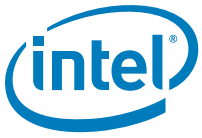
ICR bits are cleared on register read. Clear-on-read can be enabled/disabled through a general configuration register bit. Refer to Section 6.3.3 for additional information.

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 the I210-CS/CL 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.

Field	Bit(s)	Initial Value	Description
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 the I210-CS/CL writes back an Rx descriptor to memory.
Reserved	9:8	0b	Reserved. Write 0x0, ignore on read.
GPHY	10	0b	Internal 1000/100/10BASE-T PHY interrupt.
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.
Reserved	15	0b	Reserved.
Reserved	17:16	00b	Reserved. Write 0x0, ignore on read.
Reserved	18	0b	Reserved.
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.



Field	Bit(s)	Initial Value	Description
Reserved	27	0b	Reserved. Write 0x0, ignore on read.
Reserved	28	0b	Reserved.
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 the I210-CS/CL. This bit is not valid in MSI/MSI-X environments.

7.8.10 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 7.8.9). 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 7.8.11). Bits written with 0b are unchanged. Refer to Section 6.3.3 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	9:8	0b	Reserved. Write 0b, ignore on read.
GPHY	10	0b	Sets the internal 1000/100/10BASE-T 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.
Reserved	17:15	0x0	Reserved. Write 0x0, ignore on read.
Reserved	18	0b	Reserved.
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.

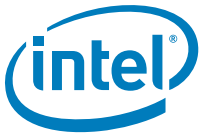
Field	Bit(s)	Initial Value	Description
Reserved	23	0b	Reserved. Write 0b, ignore on read.
PCI Exception	24	0b	Sets the PCI Exception interrupt.
SCE	25	0b	Sets the DMA Coalescing Clock Control Event interrupt.
Software WD	26	0b	Sets the Software Watchdog interrupt.
Reserved	27	0b	Reserved. Write 0b, ignore on read.
Reserved	28	0b	Reserved.
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.

7.8.11 Interrupt Mask Set/Read Register - IMS (0x1508; R/W)

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 7.8.9).

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 7.8.12) rather than writing a 0b to a bit in this register. Refer to Section 6.3.3 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	9:8	0b	Reserved. Write 0b, ignore on read.
GPHY	10	0b	Sets/Reads the mask for Internal 1000/100/10BASE-T 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.



Field	Bit(s)	Initial Value	Description
Reserved	17:15	0x0	Reserved. Write 0x0, ignore on read.
Reserved	18	0b	Reserved
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	28:27	0b	Reserved. Write 0b, ignore on read.
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.

7.8.12 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 7.8.11), and the status of the cause bit is reflected in the Interrupt Cause Read register (refer to Section 7.8.9). 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 6.3.3 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.
Reserved	5	0b	Reserved. Write 0b, ignore on read.

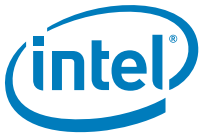
Field	Bit(s)	Initial Value	Description
Rx Miss	6	0b	Clears the mask for the Rx Miss interrupt.
RXDW	7	0b	Clears the mask for the Receiver Descriptor Write Back interrupt.
Reserved	9:8	0b	Reserved. Write 0b, ignore on read.
GPHY	10	0b	Clears the mask for the Internal 1000/100/10BASE-T PHY interrupt.
GPI_SDP0	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.
Reserved	17:15	0x0	Reserved. Write 0x0, ignore on read.
Reserved	18	0b	Reserved.
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	28:27	0b	Reserved. Write 0b, ignore on read.
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.

7.8.13 Interrupt Acknowledge Auto Mask Register - IAM (0x1510; R/W)

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 6.3.3 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.

7.8.14 Interrupt Throttle - EITR (0x1680 + 4*n [n = 0...4]; R/W)

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 the I210-CS/CL, 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), the I210-CS/CL guarantees the processor does not receive an interrupt for 125 µs from the last interrupt. The maximum observable interrupt rate from the I210-CS/CL 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.

7.8.15 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 7-23 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 7-22.

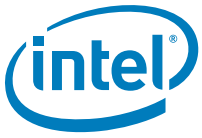
Field	Bit(s)	Initial Value	Description
INT_Alloc[4*n]	2:0	0x0	Defines the MSI-X vector assigned to Rx0 or Rx2 for IVAR[0] or IVAR[1], respectively. Valid values are 0 to 4 for MSI-X mode and 0 to 3 in non-MSI-X mode.
Reserved	6:3	0x0	Reserved. Write 0x0, ignore on read.
INT_Alloc[4*n]	7	0b	Valid bit 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.

7.8.16 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.

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.

7.8.17 General Purpose Interrupt Enable - GPIE (0x1514; RW)

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 6.3.3 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, the I210-CS/CL 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.

7.9 MSI-X Table Register Descriptions

These registers are used to configure the MSI-X mechanism. The Message Address and Message Upper Address registers set the address for each of the vectors. The message register sets the data sent to the relevant address. The vector control registers are used to enable specific vectors.

The pending bit array register indicates which vectors have pending interrupts. The structure is listed in Table 7-22.

Table 7-22. 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 (N-1)	Base + (N-1) *16

Note: N = 5.

Table 7-23. MSI-X PBA Structure

MSIXPBA[63:0]	Qword Number	BAR 3 - Offset
Pending Bits 0 through 63	QWORD0	Base (0x2000)
Pending Bits 64 through 127	QWORD1	Base+1*8
...	...	...
Pending Bits ((N-1) div 64)*64 through N-1	QWORD((N-1) div 64)	BASE + ((N-1) div 64)*8

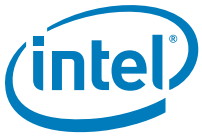
Note: N = 5. As a result, only Qword0 is implemented.

7.9.1 MSI-X Table Entry Lower Address - MSIXTADD (BAR3: 0x0000 + 0x10*n [n=0...4]; R/W)

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.

7.9.2 MSI-X Table Entry Upper Address - MSIXTUADD (BAR3: 0x0004 + 0x10*n [n=0...4]; R/W)

Field	Bit(s)	Initial Value	Description
Message Address	31:0	0x0	System-Specific Message Upper Address.



7.9.3 MSI-X Table Entry Message - MSIXTMSG (BAR3: 0x0008 + 0x10*n [n=0...4]; R/W)

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.

7.9.4 MSI-X Table Entry Vector Control - MSIXTVCTRL (BAR3: 0x000C + 0x10*n [n=0...4]; R/W)

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.

7.9.5 MSIXPBA Bit Description – MSIXPBA (BAR3: 0x2000; RO)

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.

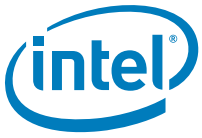
7.9.6 MSI-X PBA Clear – PBACL (0x5B68; R/W1C)

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.

7.10 Receive Register Descriptions

7.10.1 Receive Control Register - RCTL (0x0100; R/W)

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 10/100/1000BASE-T mode) in order to be recognized by the I210-CS/CL (even bad packets).
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 the I210-CS/CL can receive is defined in the *RLPML.RLPML* register.
LBM	7:6	00b	Loopback Mode. Controls the loopback mode of the I210-CS/CL. 00b = Normal operation (or PHY loopback in 10/100/1000BASE-T mode). 01b = MAC loopback (test mode). 10b = Undefined. 11b = Loopback via internal SerDes (SerDes/SGMII/KX mode only). When using the internal PHY, LBM should remain set to 00b and the PHY instead configured for loopback through the MDIO interface. ***Note:*** PHY devices require programming for loopback operation using MDIO accesses.
Reserved	11:8	0x0	Reserved. Write 0x0, ignore on read.

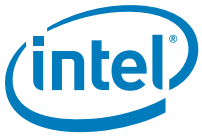


Field	Bit(s)	Initial Value	Description
MO	13:12	00b	Multicast Offset. Determines which bits of the incoming multicast address are used in looking up the bit vector. 00b = bits [47:36] of received destination multicast address. 01b = bits [46:35] of received destination multicast address. 10b = bits [45:34] of received destination multicast address. 11b = bits [43:32] of received destination multicast address.
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.

7.10.2 Split and Replication Receive Control - SRRCTL (0xC00C + 0x40*n [n=0...3]; R/W)

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.
DMACQ_Dis	7	0b	DMA Coalescing Disable. 0b= Enable DMA coalescing on this queue if *DMACR.DMAC_EN* is set to 1b. 1b = Disable DMA coalescing on this queue. When a packet is destined to this queue and the device is in coalescing mode, coalescing mode is exited immediately and PCIe moves to the L0 link power management state.
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).
Reserved	19:14	0x0	Reserved. Write 0x0, ignore on read.
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). Reserved. 111b = Reserved.



Field	Bit(s)	Initial Value	Description
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 the I210-CS/CL, 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.

7.10.3 Packet Split Receive Type - PSRTYPE (0x5480 + 4*n [n=0...3]; R/W)

This register enables or disables each type of header that needs to be split or replicated (refer to Section 6.1.5 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.

7.10.4 Receive Descriptor Base Address Low - RDBAL (0xC000 + 0x40*n [n=0...3]; R/W)

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.

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x2800, 0x2900, 0x2A00 and 0x2B00, respectively.

Field[1]	Bit(s)	Initial Value	Description
Lower_0	6:0	0x0	Ignored on writes. Returns 0x0 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).

7.10.5 Receive Descriptor Base Address High - RDBAH (0xC004 + 0x40*n [n=0...3]; R/W)

This register contains the upper 32 bits of the 64-bit descriptor base address.

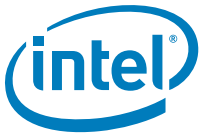
Field[1]	Bit(s)	Initial Value	Description
RDBAH	31:0	X	Receive Descriptor Base Address [63:32].

1. Software should program the *RDBAH[n]* register only when a queue is disabled (*RXDCTL[n].Enable* = 0b).

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x2804, 0x2904, 0x2A04 and 0x2B04, respectively.

7.10.6 Receive Descriptor Ring Length - RDLEN (0xC008 + 0x40*n [n=0...3]; R/W)

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).

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x2808, 0x2908, 0x2A08 and 0x2B08, respectively.

7.10.7 Receive Descriptor Head - RDH (0xC010 + 0x40*n [n=0...3]; RO)

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.

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x2810, 0x2910, 0x2A10 and 0x2B10, respectively.

7.10.8 Receive Descriptor Tail - RDT (0xC018 + 0x40*n [n=0...3]; R/W)

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 the I210-CS/CL.

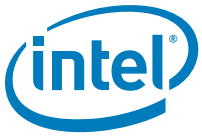
In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x2818, 0x2918, 0x2A18 and 0x2B18, respectively.

Field	Bit(s)	Initial Value	Description
RDT	15:0	0x0	Receive Descriptor Tail.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

7.10.9 Receive Descriptor Control - RXDCTL (0xC028 + 0x40*n [n=0...3]; R/W)

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 the I210-CS/CL 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.
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 the I210-CS/CL. ***Note:*** It's recommended not to place a value above 0xC in the *WTHRESH* field.
Reserved	24:21	0x0	Reserved.
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.



Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x2828, 0x2928, 0x2A28 and 0x2B28, respectively.

7.10.10 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.

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x2830, 0x2930, 0x2A30 and 0x2B30, respectively.

Packets dropped due to the queue being disabled might not be counted by this register.

7.10.11 Transmit Queue Drop Packet Count - TQDPC (0xE030 + 0x40*n [n=0...3]; RW)

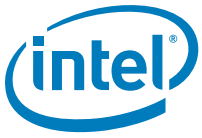
Field	Bit(s)	Initial Value	Description
TQDPC	31:0	0x0	Transmit Queue Drop Packet Count. Counts the number of packets dropped by a queue due to lack of space in the loopback buffer or due to security (anti-spoof) issues. A multicast packet dropped by some of the destinations, but sent to others is counted by this counter. ***Note:*** Counter wraps around when reaching a value of 0xFFFFFFFF.

7.10.12 Receive Checksum Control - RXCSUM (0x5000; R/W)

The Receive Checksum Control register controls the receive checksum off loading features of the I210-CS/CL. The I210-CS/CL 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 6.1.7.2 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, the I210-CS/CL 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, the I210-CS/CL calculates the TCP or UDP checksum and indicates a pass/fail indication to software via 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, the I210-CS/CL 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.
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.



7.10.13 Receive Long Packet Maximum Length - RLPML (0x5004; R/W)

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.

7.10.14 Receive Filter Control Register - RFCTL (0x5008; R/W)

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.
Reserved	10	0b	Reserved.
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.
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.
Reserved	31:20	0x0	Reserved. Write 0x0, ignore on read.

7.10.15 Multicast Table Array - MTA (0x5200 + 4*n [n=0...127]; R/W)

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 7-1 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.

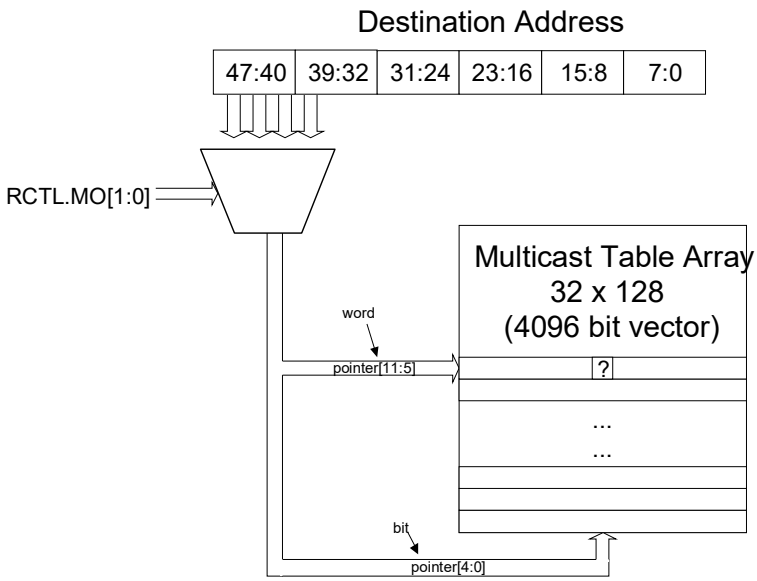


Figure 7-1. Multicast Table Array

7.10.16 Receive Address Low - RAL (0x5400 + 8*n [n=0...15]; R/W)

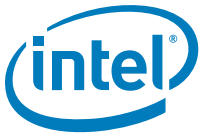
While "n" is the exact unicast/multicast address entry and it is equal to 0,1,...15.

These registers contain the lower bits of the 48 bit Ethernet address. All 32 bits are valid.

These registers are reset by a software reset or platform reset. If a Flash is present, the first register (*RAL0*) is loaded from the Flash after a software or platform reset.

Note: The *RAL* field should be written in network order.

Field	Bit(s)	Initial Value	Description
RAL	31:0	X	Receive Address Low. Contains the lower 32-bit of the 48-bit Ethernet address.



7.10.17 Receive Address High - RAH (0x5404 + 8*n [n=0…15]; R/W)

These registers contain the upper bits of the 48-bit Ethernet address. The complete address is [*RAH*, *RAL*]. The *RAH.AV* bit determines whether this address is compared against the incoming packet.

The *RAH.ASEL* field enables the I210-CS/CL to perform special filtering on receive packets.

After reset, if an Flash is present, the first register (Receive Address Register 0) is loaded from the *IA* field in the Flash with its *Address Select* field set to 00b and its *Address Valid* field set to 1b. If no Flash is present, the *Address Valid* field is set to 0b and the *Address Valid* field for all of the other registers is set to 0b.

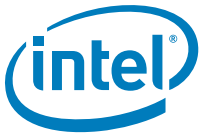
Note: The *RAH* field should be written in network order.

The first receive address register (*RAH[0]*) is also used for exact match pause frame checking (DA matches the first register). As a result, *RAH[0]* should always be used to store the individual Ethernet MAC address of the I210-CS/CL.

Field	Bit(s)	Initial Value	Description
RAH	15:0	X	Receive address High. Contains the upper 16 bits of the 48-bit Ethernet address.
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. In Qav mode, indicates which Rx queue should get the packets matching this MAC address. This field maps to the relevant queue: 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 a Flash is present, the *Address Valid* field of the Receive Address Register 0 is set to 1b after a software or PCI reset or Flash read. In entries 0-15 this bit is cleared by master reset.

7.10.18 VLAN Priority Queue Filter VLAPQF (0x55B0;R/W)

Field	Bit(s)	Initial Value	Description
VP0QSEL	1:0	0x0	VLAN Priority 0 Queue Selection. This field defines the target queue for packets with VLAN priority value of 0x0 and are enabled by VLANPV.
Reserved	2	0x0	Reserved.
VLANP0V	3	0x0	VLAN Priority 0 Valid. This field enables VLAN Priority 0x0 for queue selection.
VP1QSEL	5:4	0x0	VLAN Priority 1Queue Selection. This field defines the target queue for packets with VLAN priority value of 0x1 and are enabled by VLANPV.
Reserved	6	0x0	Reserved.
VLANP1V	7	0x0	VLAN Priority 1 Valid. This field enables VLAN Priority 0x1 for queue selection.
VP2QSEL	9:8	0x0	VLAN Priority 2 Queue Selection. This field defines the target queue for packets with VLAN priority value of 0x2 and are enabled by VLANPV.
Reserved	10	0x0	Reserved.
VLANP2V	11	0x0	VLAN Priority 2 Valid. This field enables VLAN Priority 0x2 for queue selection.
VP3QSEL	13:12	0x0	VLAN Priority 3 Queue Selection. This field defines the target queue for packets with VLAN priority value of 0x3 and are enabled by VLANPV.
Reserved	14	0x0	Reserved.
VLANP3V	15	0x0	VLAN Priority 3 Valid. This field enables VLAN Priority 0x3 for queue selection.
VP4QSEL	17:16	0x0	VLAN Priority 4 Queue Selection. This field defines the target queue for packets with VLAN priority value of 0x4 and are enabled by VLANPV.
Reserved	18	0x0	Reserved.
VLANP4V	19	0x0	VLAN Priority 4 Valid. This field enables VLAN Priority 4 for queue selection.
VP5QSEL	21:20	0x0	VLAN Priority 5 Queue Selection. This field defines the target queue for packets with VLAN priority value of 0x5 and are enabled by VLANPV.
Reserved	22	0x0	Reserved.
VLANP5V	23	0x0	VLAN Priority 5 Valid. This field enables VLAN Priority 0x5 for queue selection.
VP6QSEL	25:24	0x0	VLAN Priority 6 Queue Selection. This field defines the target queue for packets with VLAN priority value of 0x6 and are enabled by VLANPV.
Reserved	26	0x0	Reserved.
VLANP6V	27	0x0	VLAN Priority 6 Valid. This field enables VLAN Priority 0x6 for queue selection.
VP7QSEL	29:28	0x0	VLAN Priority 7 Queue Selection. This field defines the target queue for packets with VLAN priority value of 0x7 and are enabled by VLANPV.
Reserved	30	0x0	Reserved.
VLANP7V	31	0x0	VLAN Priority 7 Valid. This field enables VLAN Priority 0x7 for queue selection.



7.10.19 VLAN Filter Table Array - VFTA (0x5600 + 4*n [n=0...127]; R/W)

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 7.10.15 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.

7.10.20 Multiple Receive Queues Command Register - MRQC (0x5818; R/W)

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.

Field	Bit(s)	Initial Value	Description
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. 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.

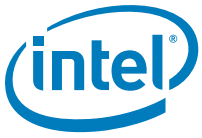
Note: The *MRQC.Multiple Receive Queues Enable* field is used to enable/disable RSS hashing and also to enable multiple receive queues. Disabling this feature is not recommended. Model usage is to reset the I210-CS/CL after disabling the RSS.

7.10.21 RSS Random Key Register - RSSRK (0x5C80 + 4*n [n=0...9]; R/W)

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]



7.10.22 Redirection Table - RETA (0x5C00 + 4*n [n=0...31]; R/W)

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 R/W registers.

Field	Bit(s)	Initial Value	Description
Entry 0	7:0	0x0	Determines the tag value and physical queue for index 4*n+0 (n=0...31).
Entry 1	15:8	0x0	Determines the tag value and physical queue for index 4*n+1 (n=0...31).
Entry 2	23:16	0x0	Determines the tag value and physical queue for index 4*n+2 (n=0...31).
Entry 3	31:24	0x0	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.

7.10.23 DMA VM Offload Register - DVMOLR (0xC038 + 0x40*n[n=0...3]; RW)

This register controls part of the offload and queueing options applied to each queue.

Field	Bit(s)	Initial Value	Description
Reserved	28:0	0x0	Reserved. Write 0x0, ignore on read.
Hide VLAN	29	0b	If this bit is set, a value of zero is written in the *RDESC.VLAN tag* and in the *RDESC.STATUS.VP* fields of the received descriptor. If this bit is set for a queue, the *DVMOLR.STRVLAN* bit for this queue should be set also.
STRVLAN	30	0b	VLAN Strip. If this bit is set, the VLAN is removed from the packet, and can be inserted in the receive descriptor (depending on the value of the *Hide VLAN* field). ***Note:*** If this bit is set the *DVMOLR[n].CRC strip* bit should be set as the CRC is not valid anymore.
CRC Strip	31	1b	CRC Strip. If this bit is set, the CRC is removed from the packet. ***Notes:*** 1. If the *DVMOLR[n].STRVLAN* bit is set the *DVMOLR[n].CRC strip* 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).

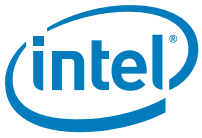
7.11 Filtering Register Descriptions

7.11.1 Immediate Interrupt RX - IMIR (0x5A80 + 4*n [n=0...7]; R/W)

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.

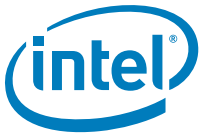
7.11.2 Immediate Interrupt Rx Ext. - IMIREXT (0x5AA0 + 4*n [n=0...7]; R/W)

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. When *DVMOLR.CRC* strip and *DVMOLR.STRVLAN* are used, the Size_thresh should include the VLAN and the CRC.
Size_BP	12	X	Size Bypass. When 1b, the size check is bypassed. When 0b, the size check is performed.

Field	Bit(s)	Initial Value	Description
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.

7.11.3 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	Mask bits for the 2-tuple fields. The corresponding field participates in the match if the following bit cleared: Bit 28 = Mask protocol comparison. Bits 31:29 = Reserved.



7.11.4 Immediate Interrupt Rx VLAN Priority - IMIRVP (0x5AC0; R/W)

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.

7.11.5 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.

7.11.6 EType Queue Filter - ETQF (0x5CB0 + 4*n[n=0...7]; RW)

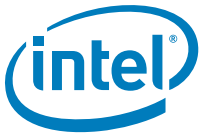
Field	Bit(s)	Initial Value	Description
EType	15:0	0x0	Identifies the protocol running on top of IEEE 802. Used to forward Rx packets containing this EType to a specific Rx queue.
Rx Queue	18:16	0x0	Identifies the receive queue associated with this EType. Valid values are 0 to 3.
Reserved	19	0x0	Reserved. 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.

Field	Bit(s)	Initial Value	Description
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.

7.12 Transmit Register Descriptions

7.12.1 Transmit Control Register - TCTL (0x0400; R/W)

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.
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.
SWXOFF	22	0b	Software XOFF Transmission. When set to 1b, the I210-CS/CL 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 the I210-CS/CL. Software should not write a 1b to this bit while the I210-CS/CL is configured for half-duplex operation.



Field	Bit(s)	Initial Value	Description
Reserved	23	0b	Reserved.
RTLC	24	0b	Re-transmit on Late Collision. When set, enables the I210-CS/CL to re-transmit on a late collision event. ***Note:*** RTLC configures the I210-CS/CL to perform re-transmission of packets when a late collision is detected. Note that the collision window is speed dependent: 64 bytes for 10/100 Mb/s and 512 bytes for 1000 Mb/s operation. 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	31:25		Reserved.

7.12.2 Transmit Control Extended - TCTL_EXT (0x0404; R/W)

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.

7.12.3 Transmit IPG Register - TIPG (0x0410; R/W)

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. Measured in increments of the MAC clock: 8 ns MAC clock when operating @ 1 Gb/s. 80 ns MAC clock when operating @ 100 Mb/s. 800 ns MAC clock when operating @ 10 Mb/s. IPGT specifies the IPG length for back-to-back transmissions in both full duplex and half duplex. Note that an offset of 4 byte times is added to the programmed value to determine the total IPG. As a result, a value of 8 is recommended to achieve a 12 byte time IPG.
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). Measured in increments of the MAC clock: 8 ns MAC clock when operating @ 1 Gb/s. 80 ns MAC clock when operating @ 100 Mb/s 800 ns MAC clock when operating @ 10 Mb/s.
IPGR	29:20	0x06	IPG After Deferral. Specifies the total IPG time for non back-to-back transmissions (transmission following deferral) in half duplex. Measured in increments of the MAC clock: 8 ns MAC clock when operating @ 1 Gb/s. 80 ns MAC clock when operating @ 100 Mb/s 800 ns MAC clock when operating @ 10 Mb/s. 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.

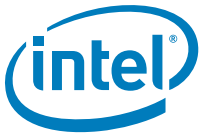
7.12.4 Retry Buffer Control – RETX_CTL (0x041C; RW)

This register controls the collision retry buffer.

Field	Bit(s)	Initial Value	Description
Water Mark	3:0	0x3	Retry buffer water mark. This parameters defines the minimal number of Qwords that should be present in the retry buffer before transmission is started.
Reserved	31:4	0x0	Reserved. Write 0x0, ignore on read.

7.12.5 DMA TX Control - DTXCTL (0x3590; R/W)

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.
Reserved	3	0x0	Reserved. Write 0x0, ignore on read.
OutOfSyncDisable	4	0b	Disable Out Of Sync Mechanism. 0b = Out Of Sync mechanism is enabled. 1b = Out Of Sync mechanism is disabled.
Reserved	6:5	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.

7.12.6 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 6.2.4.7.1 and Section 6.2.4.7.2 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.

7.12.7 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 6.2.4.7.3 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.

7.12.8 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

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	31:12	0x0	Reserved.

7.12.9 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	Maximum transmit packet size that is allowed to be transmitted by the driver. Value entered is in 64 Bytes resolution. ***Notes:*** 1. Default value enables transmission of maximum sized 9,728-byte Jumbo frames. 2. Values programmed in this field should not exceed 9,728 bytes. 3. Value programmed should not exceed the Tx buffers size programmed in the *TXPBSIZE* register.
Reserved	31:9	0x0	Reserved. Write 0x0, ignore on read.

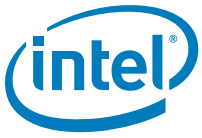
7.12.10 Transmit Descriptor Base Address Low - TDBAL (0xE000 + 0x40*n [n=0...3]; R/W)

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. Returns 0x0 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).

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x3800, 0x3900, 0x3A00 and 0x3B00, respectively.



7.12.11 Transmit Descriptor Base Address High - TDBAH (0xE004 + 0x40*n [n=0...3]; R/W)

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).

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x3804, 0x3904, 0x3A04 and 0x3B04, respectively.

7.12.12 Transmit Descriptor Ring Length - TDLEN (0xE008 + 0x40*n [n=0...3]; R/W)

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).

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x3808, 0x3908, 0x3A08 and 0x3B08, respectively.

7.12.13 Transmit Descriptor Head - TDH (0xE010 + 0x40*n [n=0...3]; RO)

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.

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x3810, 0x3910, 0x3A10 and 0x3B10, respectively.

7.12.14 Transmit Descriptor Tail - TDT (0xE018 + 0x40*n [n=0...3]; R/W)

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.

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x3818, 0x3918, 0x3A18 and 0x3B18, respectively.

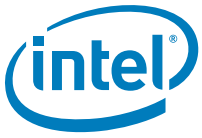
7.12.15 Transmit Descriptor Control - TXDCTL (0xE028 + 0x40*n [n=0...3]; R/W)

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 the I210-CS/CL 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 the I210-CS/CL.
Reserved	23:21	0x0	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.DataFetchARB 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 7.2.4 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.

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x3828, 0x3928, 0x3A28 and 0x3B28, respectively.

7.12.16 Tx Descriptor Completion Write-Back Address Low - TDWBAL (0xE038 + 0x40*n [n=0...3]; R/W)

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).

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x3838, 0x3938, 0x3A38 and 0x3B38, respectively.

7.12.17 Tx Descriptor Completion Write-Back Address High - TDWBAH (0xE03C + 0x40*n [n=0...3];R/W)

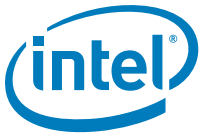
Field[1]	Bit(s)	Initial Value	Description
HeadWB_High	31:0	0x0	Highest 32 bits of the head write-back memory location.

1. Software should program the TDWBAH[n] register only when a queue is disabled (*TXDCTL[n].Enable* = 0b).

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x383C, 0x393C, 0x3A3C and 0x3B3C, respectively.

7.12.18 Tx Qav Hi Credit TQAVHC (0x300C+ 0x40*n [n=0...1];R/W)

Field	Bit(s)	Initial Value	Description
HiCredit	31:0	0x0	Hi Credit Value. Maximum number of credits that this queue can accumulate. See Section 6.2.7.6 for a description of how this field should be calculated. Relevant only if *TransmitMode* is set to 1b (Qav).



7.12.19 Tx Qav Credit Control TQAVCC (0x3004 + 0x40*n [n=0...1]; R/W)

Field	Bit(s)	Initial Value	Description
IdleSlope	15:0	0x0	IdleSlope. Idle Slope for this queue Value in credits. Must be smaller than LinkRate = 0x7735 credits/byte. See Section 6.2.7.6 for a description of how this field should be calculated. Relevant only if *TransmitMode* is set to 1b (Qav).
Reserved	29:16	0x0	Reserved.
Reserved	30	0x0	Reserved.
QueueMode	31	0x0	Queue Mode. 0b = Strict Priority. 1b = Stream Reservation. ***Note:*** Queue0 QueueMode must be set to 1b when *TransmitMode* is set to Qav. Relevant only if *TransmitMode* is set to 1b (Qav).

7.12.20 Launch Time Offset Register LAUNCH_OS0 (0x3578; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	4:0	0x0	Reserved.
LaunchOffset	29:5	0x0	Launch Time Offset, defined in 32nsec granularity. The launch time of a packet is defined by the sum of the LaunchOffset and the Relative LaunchTime parameter in the transmit context descriptor. Note that the calculated launch time should not exceed 1 second on which SYSTIML wraps around.
Reserved	30	0x0	Reserved.
Reserved	31	0x1	Reserved, should be written to 0x1 and ignored on read.

7.12.21 Tx Qav Control TQAVCTRL (0x3570; R/W)

Field	Bit(s)	Initial Value	Description
TransmitMode	0	0b	Transmit Mode Configuration. 0b= Legacy. 1b= Qav. ***Note:*** Any change to this field shall be done while the queue is disabled.
Reserved	1	0b	Reserved.
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.
Reserved	3	0b	Reserved.
DataFetchARB	4	0x0	Data Fetch Arbitration. 0b= Round Robin. 1b= Most Empty. Relevant only if *TransmitMode* is set to 1 (Qav).
Reserved	7:5	0x0	Reserved.

Field	Bit(s)	Initial Value	Description
DataTranARBsd	8	0x0	Data Transmit Arbitration. 0b = Strict Priority. 1b = Credit Shaper Algorithm. Relevant only if *TransmitMode* is set to 1b (Qav).
DataTranTIM	9	0x0	Data Launch Time Valid. Relevant only if *TransmitMode* is set to 1b (Qav).
SP_WAIT_SR	10	0x0	When set to 1b, the SP queues wait for the SR queues to make sure the SR launch time is always guaranteed.
Reserved	15:11	0x0	Reserved.
FetchTimDelta	31:16	0x0	Fetch Time Delta. This field holds the value to be reduced from the launch time for fetch time decision. The FetchTimeDelta value is defined in 32 ns granularity. Relevant only if *TransmitMode* is set to 1b (Qav).

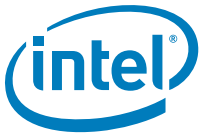
7.13 DCA and TPH Register Descriptions

7.13.1 Rx DCA Control Registers - RXCTL (0xC014 + 0x40*n [n=0...3]; R/W)

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.



Field	Bit(s)	Initial Value	Description
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.
RxRepHeader ROEn	15	1b	Receive Replicated/Split Header Relax Order Enable.
Reserved	23:16	0x0	Reserved. Write 0x0, ignore on read.
CPUID	31:24	0x0	Physical ID. Legacy DCA capable platforms. The software device driver, upon discovery of the physical CPU ID and CPU Bus ID, programs the CPUID field with the Physical CPU and Bus ID associated with this Rx queue. DCA 1.0 capable platforms. The software device driver programs a value, based on the relevant APIC ID, associated with this Tx queue. TPH capable platforms. The device driver programs a value, based on the relevant Socket ID, associated with this receive queue. Note that for TPH platforms, bits 31:27 of this field should always be set to zero. Refer to Section 7.7.2 for details.

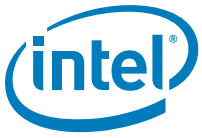
Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x2814, 0x2914, 0x2A14 and 0x2B14, respectively.

7.13.2 Tx DCA Control Registers - TXCTL (0xE014 + 0x40*n [n=0...3]; R/W)

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	23:14	0b	Reserved Write 0 ignore on read.
CPUID	31:24	0x0	Physical ID Legacy DCA capable platforms - the device driver, upon discovery of the physical CPU ID and CPU Bus ID, programs the CPUID field with the Physical CPU and Bus ID associated with this Tx queue. DCA 1.0 capable platforms - the device driver programs a value, based on the relevant APIC ID, associated with this Tx queue. TPH capable platforms - the device driver programs a value, based on the relevant Socket ID, associated with this transmit queue. Note that for TPH platforms, bits 31:27 of this field should always be set to zero. Refer to Section 7.7.2 for details.

1. Both the *DCA Enable* bit and the *TPH Enable* bit should not be set for the same type of traffic.

Note: In order to keep compatibility with previous devices, for queues 0-3, these registers are aliased to addresses 0x3814, 0x3914, 0x3A14 and 0x3B14, respectively.



7.13.3 DCA Requester ID Information - DCA_ID (0x5B70; RO)

The *DCA Requester ID* field, composed of Device ID, Bus #, and Function # is set up in MMIO space for software to program the DCA Requester ID Authentication register.

Field	Bit(s)	Initial Value	Description
Function Number	2:0	000b	Function Number. Function number assigned to the function based on BIOS/operating system enumeration.
Device Number	7:3	0x0	Device Number. Device number assigned to the function based on BIOS/operating system enumeration.
Bus Number	15:8	0x0	Bus Number. Bus number assigned to the function based on BIOS/operating system enumeration.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

7.13.4 DCA Control - DCA_CTRL (0x5B74; R/W)

This CSR is common to all functions.

Field	Bit(s)	Initial Value	Description
DCA_DIS	0	1b	DCA Disable. 0b = DCA tagging is enabled. 1b = DCA tagging is disabled.
DCA_MODE	4:1	0x0	DCA Mode. 000b = Legacy DCA is supported. The TAG field in the TLP header is based on the following coding: bit 0 is DCA enable; bits 3:1 are CPU ID). 001b = DCA 1.0 is supported. When DCA is disabled for a given message, the TAG field is 0000,0000b. If DCA is enabled, the TAG is set per queue as programmed in the relevant DCA Control register. All other values are undefined.
Reserved	8:5	0x0	Reserved. Write 0x0, ignore on read.
Desc_PH	10:9	00b	Descriptor PH. Defines the PH field used when a TPH hint is given for descriptor associated traffic (descriptor fetch, descriptor write back or head write back).
Data_PH	12:11	10b	Data PH. Defines the PH field used when a TPH hint is given for data associated traffic (Tx data read, Rx data write).
Reserved	31:13	0x0	Reserved. Write 0x0, ignore on read.

7.14 Timer Registers Description

7.14.1 Watchdog Setup - WDSTP (0x1040; R/W)

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 the I210-CS/CL 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 Flash.

1. Value read from the Flash.

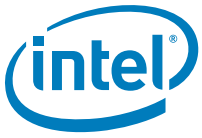
7.14.2 Watchdog Software Device Status - WDSWSTS (0x1044; R/W)

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 the I210-CS/CL is malfunctioning. The encoding of this field is software/firmware dependent. A value of 0x0 indicates a functional the I210-CS/CL.

7.14.3 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).

7.14.4 TCP Timer - TCPTIMER (0x104C; R/W)

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.

7.15 Time Sync Register Descriptions

7.15.1 Rx Time Sync Control Register - TSYNCRXCTL (0xB620;RW)

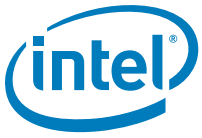
Field	Bit(s)	Initial Value	Description
RXTT(RO)	0	0x0	Rx Timestamp Valid Bit is set when a valid value for Rx timestamp is captured in the Rx timestamp registers. Bit is cleared by read of Rx timestamp high register (*RXSTMPH*)).
Type	3:1	0x0	Type of Packets to Timestamp. 000b = Timestamp L2 (V2) packets 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.
RSV	31:5	0x0	Reserved. Write 0x0, ignore on read.

7.15.2 Rx Timestamp Low - RXSTMPL (0xB624; RO)

Field	Bit(s)	Initial Value	Description
RTSL	29:0	0x0	Rx timestamp LSB value (defined in ns units).
Zero	31:30	0x0	Zero bits.

7.15.3 Rx Timestamp High - RXSTMPH (0xB628; RO)

Field	Bit(s)	Initial Value	Description
RTSH	31:0	0x0	Rx timestamp MSB value (defined in second units).



7.15.4 Tx Time Sync Control Register - TSYNCTXCTL (0xB614; RW)

Field	Bit(s)	Initial Value	Description
TXTT(ROM)	0	0b	Transmit timestamp valid (equals 1b when a valid value for Tx timestamp is captured in the Tx timestamp register, clear by read of Tx timestamp register TXSTMPH).
RSV	3:1	0x0	Reserved. Write 0x0, ignore on read.
EN	4	0b	Enable Transmit timestamp. 0b = time stamping disabled. 1b = time stamping enabled.
RSV	5:7	0x0	Reserved. Write 0x0, ignore on read.
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.

7.15.5 Tx Timestamp Value Low - TXSTMPL (0xB618;RO)

Field	Bit(s)	Initial Value	Description
TTSL	29:0	0x0	Transmit timestamp LSB value (defined in ns units).
Zero	31:30	0x0	Zero bits.

7.15.6 Tx Timestamp Value High - TXSTMPH(0xB61C; RO)

Field	Bit(s)	Initial Value	Description
TTSH	31:0	0x0	Transmit timestamp MSB value (defined in sec units).

7.15.7 System Time Register Residue - SYSTIMR (0xB6F8; RW)

Field	Bit(s)	Initial Value	Description
STR	31:0	0x0	System time Residue value (defined in 2^{-32} nS resolution).

7.15.8 System Time Register Low - SYSTIML (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.

7.15.9 System Time Register High - SYSTIMH (0xB604; RW)

Field	Bit(s)	Initial Value	Description
STH	31:0	0x0	System time MSB value (defined in sec units).

7.15.10 System Time Register Tx MS - SYSTIMTM (0xB6FC; RW)

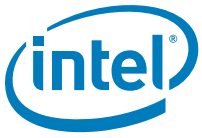
Field	Bit(s)	Initial Value	Description
STM	15:0	0x0	Two MS bytes of the system time (defined in 2^{32} sec units). This field is static, kept at the value programmed by the software. It is used for 1-step transmission as the two MS bytes inserted to the SYNC packet.
RSV	31:16	0x0	Reserved. Write 0x0, ignore on read.

7.15.11 Increment Attributes Register - TIMINCA (0xB608; RW)

Field	Bit(s)	Initial Value	Description
Incvalue	30:0	0x0	Increment value. Value to be added or subtracted (depending on ISGN value) from 8 nS clock cycle in resolution of 2^{-32} nS.
ISGN	31	0b	Increment sign. 0b = Each 8 nS cycle add to SYSTIM a value of 8 nS + Incvalue * 2^{-32} nS. 1b = Each 8 nS cycle add to SYSTIM a value of 8 nS -Incvalue * 2^{-32} nS.

7.15.12 Time Adjustment Offset Register - TIMADJ (0xB60C; RW)

Field	Bit(s)	Initial Value	Description
Tadjus	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).
Zero	30	0b	Zero bit.
Sign	31	0b	Sign (0b= "+"1b = "-").



7.15.13 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.
EN_TT1	1	0b	Enable target time 1. Enable bit is set by software to 1b, to enable a level change.
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.
SAMP_AUT0	3	0b	When setting the SAMP_AUT0 flag the SYSTIML/H registers are latched to the AUXSTMPL0/ AUXSTMPH0 registers. Then the SAMP_AUT0 flag is auto-cleared by the hardware.
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.
SAMP_AUT1	6	0b	When setting the SAMP_AUT1 flag the SYSTIML/H registers are latched to the AUXSTMPL1/ AUXSTMPH1 registers. Then the SAMP_AUT1 flag is auto-cleared by the hardware.
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.
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.

Field	Bit(s)	Initial Value	Description
Reserved	29:18	0b	Reserved. Write 0b, ignore on read.
Reserved	30	1b	Reserved. Write 1b, ignore on read.
Disable systime	31	1b	Disable SYSTIM Count Operation. 0b = SYSTIM timer activated 1b = *SYSTIM* timer disabled. Value of *SYSTIMH, SYSTIML* and *SYSTIMR* remains constant.

7.15.14 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).
Zero	31:30	0x0	Zero bits.

7.15.15 Target Time Register 0 High - TRGTTIMH0 (0xB648; RW)

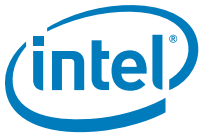
Field	Bit(s)	Initial Value	Description
TTH	31;0	0x0	Target Time 0 MSB register (defined in second units).

7.15.16 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).
Zero	31:30	0x0	Zero bits.

7.15.17 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).



7.15.18 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 ns 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.

7.15.19 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 ns 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.

7.15.20 Auxiliary Time Stamp 0 Register Low - AUXSTMPL0 (0xB65C; RO)

Field	Bit(s)	Initial Value	Description
TSTL	29:0	0x0	Auxiliary Time Stamp 0 LSB value (defined in ns units).
Zero	31:30	0x0	Zero bits.

7.15.21 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).

7.15.22 Auxiliary Time Stamp 1 Register Low AUXSTMPL1 (0xB664; RO)

Field	Bit(s)	Initial Value	Description
TSTL	29:0	0x0	Auxiliary Time Stamp 1 LSB value (defined in ns units).
Zero	31:30	0x0	Zero bits.

7.15.23 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).

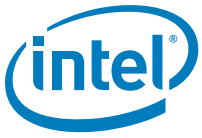
7.15.24 Time Sync RX Configuration - TSYNCRXCFG (0x5F50; R/W)

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.

7.15.25 Time Sync SDP Configuration Register - TSSDP (0x003C; R/W)

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).
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.



Field	Bit(s)	Initial Value	Description
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.

7.16 Time Sync Interrupt Registers

7.16.1 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	0b	SYSTIM Warp around. Set when SYSTIML 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.*
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.*
AUTT0	5	0b	Auxiliary Timestamp 0 Taken. Set when new timestamp is loaded into AUXSTMP 0 (auxiliary timestamp 0) register.

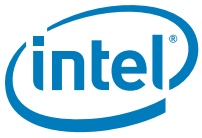
Field	Bit(s)	Initial Value	Description
AUTT1	6	0b	Auxiliary Timestamp 1 Taken. Set when new timestamp is loaded into AUXSTMP 1 (auxiliary timestamp 1) register.
TADJ	7	0b	Time Adjust Done. Set when time adjust-to-SYSTIM completes.
Reserved	31:8	0x0	Reserved. Write 0x0, ignore on read.

7.16.2 Time Sync Interrupt Mask Register - TSIM (0xB674; RW)

Field	Bit(s)	Initial Value	Description
SYS WARP	0	0b	SYSTIM Warp Around Mask. 0b = No interrupt generated when *TSICR.SWARP* is set. 1b= Interrupt generated when *TSICR.SWARP* is set.
TXTS	1	0b	Transmit Timestamp Mask. 0b = No interrupt generated when *TSICR.TXTS* is set. 1b= Interrupt generated when *TSICR.TXTS* is set.
RXTS	2	0b	Receive Timestamp Mask. 0b = No interrupt generated when *TSICR.RXTS* is set. 1b= Interrupt generated when *TSICR.RXTS* is set.
TT0	3	0b	Target time 0 Trigger Mask. 0b = No interrupt generated when *TSICR.TT0* is set. 1b= Interrupt generated when *TSICR.TT0* is set.
TT1	4	0b	Target time 1 Trigger Mask. 0b = No interrupt generated when *TSICR.TT1* is set. 1b= Interrupt generated when *TSICR.TT1* is set.
AUTT0	5	0b	Auxiliary Timestamp 0 Taken Mask. 0b = No interrupt generated when *TSICR.AUTT0* is set. 1b= Interrupt generated when *TSICR.AUTT0* is set.
AUTT1	6	0b	Auxiliary Timestamp 1 Taken Mask. 0b = No interrupt generated when *TSICR.AUTT1* is set. 1b = Interrupt generated when *TSICR.AUTT1* is set.
TADJ	7	0b	Time Adjust 0 Done Mask. 0b = No interrupt generated when *TSICR.TADJ* is set. 1b = Interrupt generated when *TSICR.TADJ* is set.
Reserved	31:8	0x0	Reserved. Write 0x0, ignore on read.

7.17 PCS Register Descriptions

These registers are used to configure the SerDes, SGMII and 1000BASE-KX PCS logic. Usage of these registers is described in Section 3.6.4.1 and Section 3.6.4.3.



7.17.1 PCS Configuration - PCS_CFG (0x4200; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	2:0	0x0	Reserved. Write 0x0, ignore on read.
PCS Enable	3	1b	PCS Enable. Enables the PCS logic of the MAC. Should be set in SGMII, 1000BASE-KX and SerDes mode for normal operation. Clearing this bit disables Rx/Tx of both data and control codes. Use this to force link down at the far end.
Reserved	29:4	0x0	Reserved. Write 0x0, ignore on read.
PCS Isolate	30	0b	PCS Isolate. Setting this bit isolates the PCS logic from the MAC's data path. PCS control codes are still sent and received.
SRESET	31	0b	Soft Reset. Setting this bit puts all modules within the MAC in reset except the Host Interface. The Host Interface is reset via HRST. This bit is NOT self clearing; GMAC is in a reset state until this bit is cleared.

7.17.2 PCS Link Control - PCS_LCTL (0x4208; RW)

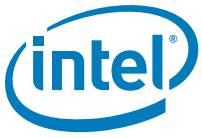
Field	Bit(s)	Initial Value	Description
FLV	0	0b	Forced Link Value. This bit denotes the link condition when force link is set. 0b = Forced link down. 1b = Forced link up.
FSV	2:1	10b	Forced Speed Value. These bits denote the speed when force speed and duplex (*PCS_LCTL.FSD*) bit is set. This value is also used when AN is disabled or when in SerDes mode. 00b = 10 Mb/s (SGMII). 01b = 100 Mb/s (SGMII). 10b = 1000 Mb/s (SerDes/SGMII/1000BASE-KX). 11b = Reserved.
FDV	3	1b	Forced Duplex Value. This bit denotes the duplex mode when force speed and duplex (*PCS_LCTL.FSD*) bit is set. This value is also used when AN is disabled or when in SerDes mode. 1b = Full duplex (SerDes/SGMII/1000BASE-KX). 0b = Half duplex (SGMII).
FSD	4	0b	Force Speed and Duplex. If this bit is set, then speed and duplex mode is forced to forced speed value and forced duplex value, respectively. Otherwise, speed and duplex mode are decided by internal AN/SYNC state machines.
Force Link	5	0b	Force Link. If this bit is set, then the internal LINK_OK variable is forced to forced link value (bit 0 of this register). Otherwise, LINK_OK is decided by internal AN/SYNC state machines.
LINK LATCH LOW (LL)	6	0b	Link Latch Low Enable. If this bit is set, then link OK going LOW (negative edge) is latched until a processor read. Afterwards, link OK is continuously updated until link OK again goes LOW (negative edge is seen).

Field	Bit(s)	Initial Value	Description
Force Flow Control	7	0b	0b = Flow control mode is set according to the AN process by following Table 37-4 in the IEEE 802.3 specification. 1b = Flow control is set according to FC_TX_EN / FC_RX_EN bits in CTRL register.
Reserved	15:8	0x0	Reserved. Write 0x0, ignore on read.
AN_ENABLE	16	0b[1]	AN Enable. Setting this bit enables the AN process in SerDes operating mode. ***Note:*** When link-up is forced (*CTRL.SLU*=1b) the AN_ENABLE bit should be 0b.
AN RESTART (SC)	17	0b	AN Restart. Used to reset/restart the link auto-negotiation process when using SerDes mode. Setting this bit restarts the auto-negotiation process. This bit is self clearing.
AN TIMEOUT EN	18	1b[1]	AN Timeout Enable. This bit enables the AN timeout feature. During AN, if the link partner does not respond with AN pages, but continues to send good IDLE symbols, then LINK UP is assumed. (This enables LINK UP condition when link partner is not AN-capable and does not affect otherwise). This bit should not be set in SGMII mode.
AN SGMII BYPASS	19	0b	AN SGMII Bypass. If this bit is set, then IDLE detect state is bypassed during AN in SGMII mode. This reduces the acknowledge time in SGMII mode.
AN SGMII TRIGGER	20	1b	AN SGMII Trigger. If this bit is cleared, then AN is not automatically triggered in SGMII mode even if SYNC fails. AN is triggered only in response to PHY messages or by a manual setting like changing the AN Enable/Restart bits.
Reserved	23:21	0x0	Reserved. Write 0x0, ignore on read.
FAST LINK TIMER	24	0b	Fast Link Timer. AN timer is reduced if this bit is set.
LINK OK FIX EN	25	1b	Link OK Fix Enable. Control for enabling/disabling LinkOK/SyncOK fix. Should be set for normal operation.
Reserved	31:26	0x0	Reserved. Write 0x0, ignore on read.

1. Bit loaded from Flash.

7.17.3 PCS Link Status - PCS_LSTS (0x420C; RO)

Field	Bit(s)	Initial Value	Description
LINK OK	0	0b	Link OK. This bit denotes the current link OK status. 0b = Link down. 1b = Link up/OK.
SPEED	2:1	10b	Speed. This bit denotes the current operating speed. 00b = 10 Mb/s. 01b = 100 Mb/s. 10b = 1000 Mb/s. 11b = Reserved.
DUPLEX	3	1b	Duplex. This bit denotes the current duplex mode. 1b = Full duplex. 0b = Half duplex.



Field	Bit(s)	Initial Value	Description
SYNC OK	4	0b	Sync OK. This bit indicates the current value of Sync OK from the PCS Sync state machine.
Reserved	15:5	0x0	Reserved. Write 0x0, ignore on read.
AN COMPLETE	16	0b	AN Complete. This bit indicates that the AN process has completed.This bit is set when the AN process reached the Link OK state. It is reset upon AN restart or reset. It is set even if the AN negotiation failed and no common capabilities were found.
AN PAGE RECEIVED	17	0b	AN Page Received. This bit indicates that a link partner's page was received during an AN process. This bit is cleared on reads.
AN TIMEDOUT	18	0b	AN Timed Out. This bit indicates an AN process was timed out. Valid after the *AN Complete* bit is set.
AN REMOTE FAULT	19	0b	AN Remote Fault. This bit indicates that an AN page was received with a remote fault indication during an AN process. This bit cleared on reads.
AN ERROR (RWM)	20	0B	AN Error. This bit indicates that a AN error condition was detected in SerDes/SGMII mode. Valid after the *AN Complete* bit is set. AN error conditions: SerDes mode: Both nodes not Full Duplex SGMII mode: PHY is set to 1000 Mb/s Half Duplex mode. Software can also force a AN error condition by writing to this bit (or can clear a existing AN error condition). This bit is cleared at the start of AN.
Reserved	31:21	0x0	Reserved. Write 0x0, ignore on read.

7.17.4 AN Advertisement - PCS_ANADV (0x4218; R/W)

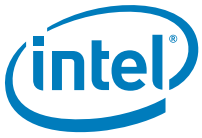
Field	Bit(s)	Initial Value	Description
Reserved	4:0	0x0	Reserved. Write 0, ignore on read.
FDCAP	5	1b	Full Duplex. Setting this bit indicates that the I210-CS/CL is capable of full duplex operation. This bit should be set to 1b for normal operation.
HDCAP (RO)	6	0b	Half Duplex. This bit indicates that the I210-CS/CL is capable of half duplex operation. This bit is tied to 0b because the I210-CS/CL does not support half duplex in SerDes mode.
ASM	8:7	00b[1]	Local PAUSE Capabilities. The I210-CS/CL's PAUSE capability is encoded in this field. 00b = No PAUSE. 01b = Symmetric PAUSE. 10b = Asymmetric PAUSE to link partner. 11b = Both symmetric and asymmetric PAUSE to the I210-CS/CL.
Reserved	11:9	0x0	Reserved. Write 0x0, ignore on read.

Field	Bit(s)	Initial Value	Description
RFLT	13:12	00b	Remote Fault. The I210-CS/CL's remote fault condition is encoded in this field. The I210-CS/CL might indicate a fault by setting a non-zero remote fault encoding and re-negotiating. 00b = No error, link OK. 01b = Link failure. 10b = Offline. 11b = Auto-negotiation error.
Reserved	14	0x0	Reserved. Write 0x0, ignore on read.
NEXTP	15	0b	Next Page Capable. The I210-CS/CL asserts this bit to request a next page transmission. The I210-CS/CL clears this bit when no subsequent next pages are requested.
Reserved	31:16	0x0	Reserved.

1. Loaded from Flash word 0x0F, bits 13:12.

7.17.5 Link Partner Ability - PCS_LPAB (0x421C; RO)

Field	Bit(s)	Initial Value	Description
Reserved	4:0	0x0	Reserved.
LPFD	5	0b	LP Full Duplex (SerDes). When set to 1b, the link partner is capable of full duplex operation. When set to 0b, the link partner is not capable of full duplex mode. This bit is reserved while in SGMII mode.
LPHD	6	0b	LP Half Duplex (SerDes). When set to 1b, the link partner is capable of half duplex operation. When set to 0b, the link partner is not capable of half duplex mode. This bit is reserved while in SGMII mode.
LPASM	8:7	00b	LP ASMDR/LP PAUSE (SerDes). The link partner's PAUSE capability is encoded in this field. 00b = No PAUSE. 01b = Symmetric PAUSE. 10b = Asymmetric PAUSE to link partner. 11b = Both symmetric and asymmetric PAUSE to the I210-CS/CL. These bits are reserved while in SGMII mode.
Reserved	9	0b	Reserved. Write 0b, ignore on read.
SGMII SPEED	11:10	00b	SerDes: Reserved. Speed (SGMII): Speed indication from the PHY.
PRF	13:12	00b	LP Remote Fault (SerDes). The link partner's remote fault condition is encoded in this field. 00b = No error, link OK. 10b = Link failure. 01b = Offline. 11b = Auto-negotiation error. SGMII [13]: Reserved. SGMII [12]: Duplex mode indication from the PHY.



Field	Bit(s)	Initial Value	Description
ACK	14	0b	Acknowledge (SerDes). The link partner has acknowledged receiving a page. SGMII: Reserved.
LPNEXTP	15	0b	LP Next Page Capable (SerDes). The link partner asserts this bit to indicate its ability to accept next pages. SGMII: Link OK indication from the PHY.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

7.17.6 Next Page Transmit - PCS_NPTX (0x4220; RW)

Field	Bit(s)	Initial Value	Description
CODE	10:0	0x0	Message/Unformatted Code Field. The *Message* field is an 11-bit wide field that encodes 2048 possible messages. The *Unformatted Code* field is an 11-bit wide field that might contain an arbitrary value.
TOGGLE	11	0b	Toggle. This bit is used to ensure synchronization with the pink partner during next page exchange. This bit always takes the opposite value of the *Toggle* bit in the previously exchanged Link Code word. The initial value of the *Toggle* bit in the first next page transmitted is the inverse of bit 11 in the base Link Code word and, therefore, can assume a value of 0b or 1b. The *Toggle* bit is set as follows: 0b = Previous value of the transmitted Link Code word when 1b. 1b = Previous value of the transmitted Link Code word when 0b.
ACK2	12	0b	Acknowledge 2. Used to indicate that a device has successfully received its Link Partners' Link Code word.
PGTYPE	13	0b	Message/Unformatted Page. This bit is used to differentiate a message page from an unformatted page. The encoding is: 0b = Unformatted page. 1b = Message page.
Reserved	14	-	Reserved. Write 0b, ignore on read.
NXTPG	15	0b	Next Page. Used to indicate whether or not this is the last next page to be transmitted. The encoding is: 0b = Last page. 1b = Additional next pages follow.
Reserved	31:16	-	Reserved. Write 0x0, ignore on read.

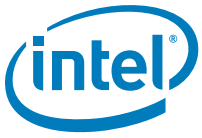
7.17.7 Link Partner Ability Next Page - PCS_LPABNP (0x4224; RO)

Field	Bit(s)	Initial Value	Description
CODE	10:0	-	Message/Unformatted Code Field. The *Message* field is an 11-bit wide field that encodes 2048 possible messages. The *Unformatted Code* field is an 11-bit wide field that might contain an arbitrary value.
TOGGLE	11	-	Toggle. This bit is used to ensure synchronization with the link partner during next page exchange. This bit always takes the opposite value of the *Toggle* bit in the previously exchanged Link Code word. The initial value of the *Toggle* bit in the first next page transmitted is the inverse of bit 11 in the base Link Code word and, therefore, can assume a value of 0b or 1b. The *Toggle* bit is set as follows: 0b = Previous value of the transmitted Link Code word when 1b. 1b = Previous value of the transmitted Link Code word when 0b.
ACK2	12	-	Acknowledge 2. Used to indicate that a device has successfully received its Link Partners' Link Code word.
MSGPG	13	-	Message Page. This bit is used to differentiate a message page from an unformatted page. The encoding is: 0b = Unformatted page. 1b = Message page.
ACK	14	-	Acknowledge. The link partner has acknowledged receiving a next page.
NXTPG	15	-	Next Page. Used to indicate whether or not this is the last next page to be transmitted. The encoding is: 0b = Last page. 1b = Additional Next Pages follow.
Reserved	31:16	-	Reserved. Write 0x0, ignore on read.

7.17.8 SFP I^2C Command- I2CCMD (0x1028; R/W)

This register is used by software or firmware to read or write to the configuration registers in an SFP module when either the *CTRL_EXT.I2C Enabled* or the *CTRL_EXT.I2C over SDP Enabled* bit is set to 1b. Prior to write accessing this register, the I^2C semaphore ownership must be taken, and released at the end of the access sequence.

Note: According to the SFP specification, only reads are allowed from this interface; however, SFP vendors also provide a writable register through this interface (for example, PHY registers). As a result, write capability is also supported.



Field	Bit(s)	Initial Value	Description
DATA	15:0	X	Data. In a Write command, software places the data bits and then the MAC shifts them out to the I^2C bus. In a Read command, the MAC reads these bits serially from the I^2C bus and then software reads them from this location. ***Note:*** This field is read in byte order and not in word order.
REGADD	23:16	0x0	I^2C Register Address. For example, register 0, 1, 2... 255.
PHYADD	26:24	0x0	Device Address Bits 3 -1. The actual address used is b{1010, PHYADD[2:0], 0}. On power up, FW loads a default value from the PHY_ADD field in NVM Initialization Control 4 (word 0x13).
OP	27	0b	Op-code. 0b = I^2C write. 1b = I^2C read.
Reset	28	0b	Reset Sequence. If set, sends a reset sequence before the actual read or write. This bit is self clearing. A reset sequence is defined as nine consecutive stop conditions.
R	29	0b	Ready Bit. Set to 1b by the I210-CS/CL at the end of the I^2C transaction. For example, indicates a read or write completed. Reset by a software write of a command.
I	30	0b	Interrupt Enable. When set to 1b by software, it causes an interrupt to be asserted to indicate the end of an I^2C cycle (*ICR.MDAC*).
E	31	0b	Error. This bit set is to 1b by hardware when it fails to complete an I^2C read. Reset by a software write of a command. ***Note:*** Bit is valid only when *Ready* bit is set.

7.17.9 SFP I2C Parameters - I2CPARAMS (0x102C; R/W)

This register is used to set the parameters for the I^2C access to the SFP module and to enable bit-banging access to the I^2C interface, when either the *CTRL_EXT.I2C Enabled* or the *CTRL_EXT.I2C over SDP Enabled* bit is set to 1b. Prior to write accessing this register, the I^2C semaphore ownership must be taken, and released at the end of the access sequence.

Field	Bit(s)	Initial Value	Description
Write Time	4:0	110b	Write Time. Defines the delay between a write access and the next access. The value is in milliseconds. A value of zero is not valid.
Read Time	7:5	010b	Read Time. Defines the delay between a read access and the next access. The value is in microseconds. A value of Zero is not valid
I2CBB_EN	8	0b	I^2C Bit-bang Enable. If set, the I^2C_CLK and I^2C_DATA lines are controlled via the CLK, DATA and DATA_OE_N fields of this register. Otherwise, they are controlled by the hardware machine activated via the I2CCMD or MDIC registers.
CLK	9	0b	I^2C Clock. While in bit-bang mode, controls the value driven on the I2C_CLK pad.

Field	Bit(s)	Initial Value	Description
DATA_OUT	10	0b	I^2C_DATA. While in bit-bang mode and when the DATA_OE_N field is zero, controls the value driven on the I2C_DATA pad.
DATA_OE_N	11	0b	I^2C_DATA_OE_N. While in bit-bang mode, controls the direction of the I2C_DATA pad. 0b = Pad is output. 1b = Pad is input.
DATA_IN (RO)	12	X	I^2C_DATA_IN. Reflects the value of the I2C_DATA pad. While in bit-bang mode when the DATA_OE_N field is zero, this field reflects the value set in the DATA_OUT field.
CLK_OE_N	13	0b	I^2C Clock Output Enable. While in bit-bang mode, controls the direction of the I2C_CLK pad. 0b = Pad is output. 1b = Pad is input.
CLK_IN (RO)	14	X	I^2C Clock In Value. Reflects the value of the I2C_CLK pad. While in bit-bang mode when the CLK_OE_N field is zero, this field reflects the value set in the CLK_OUT field.
clk_stretch_dis	15	0b	0b = Enable slave clock stretching support in I^2C access. 1b = Disable clock stretching support in I^2C access.
Reserved	31:16	0x0	Reserved. Write 0x0, ignore on read.

7.18 Statistics Register Descriptions

All Statistics registers reset when read. In addition, they stick at 0xFFFF_FFFF when the maximum value is reached.

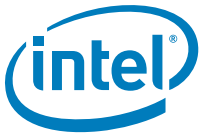
For the receive statistics it should be noted that a packet is indicated as received if it passes the I210-CS/CL's filters and is placed into the packet buffer memory. A packet does not have to be transferred to host memory in order to be counted as received.

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.

7.18.1 CRC Error Count - CRCERRS (0x4000; RC)

Counts the number of receive packets with CRC errors. 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>, inclusively) in length. If receives are not enabled, then this register does not increment.

Field	Bit(s)	Initial Value	Description
CEC	31:0	0x0	CRC Error Count.



7.18.2 Alignment Error Count - ALGNERRC (0x4004; RC)

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 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. This register is valid only in MII mode during 10/100 Mb/s operation.

Field	Bit(s)	Initial Value	Description
AEC	31:0	0x0	Alignment Error Count.

7.18.3 Symbol Error Count - SYMERRS (0x4008; RC)

Counts the number of symbol errors between reads. The count increases for every bad symbol received, whether or not a packet is currently being received and whether or not the link is up. When working in SerDes/SGMII/1000BASE-KX mode these statistics can be read from the SCVPC register.

Field	Bit(s)	Initial Value	Description
SYMERRS	31:0	0x0	Symbol Error Count.

7.18.4 RX Error Count - RXERRC (0x400C; RC)

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 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.

This register is not available in SerDes/SGMII/1000BASE-KX modes.

Field	Bit(s)	Initial Value	Description
RXEC	31:0	0x0	Rx Error Count

7.18.5 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.

7.18.6 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 the I210-CS/CL is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
SCC	31:0	0x0	Number of times a transmit encountered a single collision.

7.18.7 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 the I210-CS/CL is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
ECC	31:0	0x0	Number of packets with more than 16 collisions.

7.18.8 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 the I210-CS/CL is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
MCC	31:0	0x0	Number of times a successful transmit encountered multiple collisions.

7.18.9 Late Collisions Count - LATECOL (0x4020; RC)

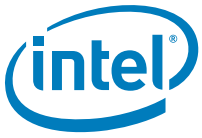
Late collisions are collisions that occur after one slot time. This register only increments if transmits are enabled (*TCTL.EN* is set) and the I210-CS/CL is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
LCC	31:0	0x0	Number of packets with late collisions.

7.18.10 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 the I210-CS/CL is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
CCC	31:0	0x0	Total number of collisions experienced by the transmitter.



7.18.11 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.

7.18.12 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 Mbps half-duplex mode .

Field	Bit(s)	Initial Value	Description
TNCRS	31:0	0x0	Number of transmissions without a CRS assertion from the PHY.

7.18.13 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.

7.18.14 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.

7.18.15 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.

7.18.16 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.

7.18.17 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.

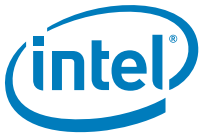
7.18.18 XOFF Transmitted Count - XOFFTXC (0x4054; RC)

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.

7.18.19 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't match standard address filtering.

Field	Bit(s)	Initial Value	Description
FCRUC	31:0	0x0	Number of unsupported flow control frames received.

7.18.20 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. 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.

7.18.21 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. 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.

7.18.22 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. 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.

7.18.23 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. 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.

7.18.24 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. 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.

7.18.25 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 7.18.37). 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).

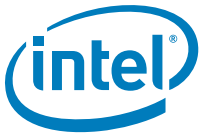
Due to changes in the standard for maximum frame size for VLAN tagged frames in 802.3, the I210-CS/CL accepts packets that have a maximum length of 1522 bytes. The RMON statistics associated with this range has been extended to count 1522 byte long packets. If *CTRL_EXT.EXT_VLAN* is set, packets up to 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.

7.18.26 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 7.18.37). 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.

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).

7.18.27 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.

Field	Bit(s)	Initial Value	Description
BPRC	31:0	0x0	Number of broadcast packets received.

7.18.28 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.

Field	Bit(s)	Initial Value	Description
MPRC	31:0	0x0	Number of multicast packets received.

7.18.29 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.

7.18.30 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. 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 - lower 4 bytes.

7.18.31 Good Octets Received Count - GORCH (0x408C; RC)

Field	Bit(s)	Initial Value	Description
GORCH	31:0	0x0	Number of good octets received - upper 4 bytes.

7.18.32 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.

7.18.33 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.

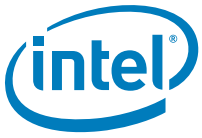
7.18.34 Receive No Buffers Count - RNBC (0x40A0; RC)

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.



7.18.35 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).

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.

7.18.36 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).

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.

7.18.37 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.

Field	Bit(s)	Initial Value	Description
ROC	31:0	0x0	Number of receive oversize errors.

7.18.38 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.

Field	Bit(s)	Initial Value	Description
RJC	31:0	0x0	Number of receive jabber errors.

7.18.39 BMC2OS Packets Received by Host - B2OGPRC (0x4158; RC)

This register counts the total number of packets originating 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.

7.18.40 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.

7.18.41 OS2BMC Packets Transmitted by Host - O2BSPC (0x415C; RC)

This register counts the total number of packets originating from the function that were sent to This includes packets received by and packets dropped in the I210-CS/CL 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.

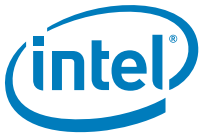
Field	Bit(s)	Initial Value	Description
OS2BSPC	31:0	0x0	OS2BMC good packets transmit count.

7.18.42 Total Octets Received - TORL (0x40C0; RC)

These registers make up a logical 64-bit register that counts the total number of octets received. This register must be accessed using two independent 32-bit accesses; TORL must be read before TORH. This register sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached.

All packets received have their octets summed into this register, regardless of their length, whether they are erred, or whether they are flow control packets. This register includes bytes received in a packet from the <Destination Address> field through the <CRC> field, inclusive. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Note: Broadcast rejected packets are counted in this counter (as opposed to all other rejected packets that are not counted).



Field	Bit(s)	Initial Value	Description
TORL	31:0	0x0	Number of total octets received - lower 4 bytes.

7.18.43 Total Octets Received - TORH (0x40C4; RC)

Field	Bit(s)	Initial Value	Description
TORH	31:0	0x0	Number of total octets received - upper 4 bytes.

7.18.44 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.

7.18.45 Total Octets Transmitted - TOTH (0x40CC; RC)

Field	Bit(s)	Initial Value	Description
TOTH	31:0	0x0	Number of total octets transmitted - upper 4 bytes.

7.18.46 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.

7.18.47 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.

7.18.48 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.

7.18.49 Packets Transmitted [65-127 Bytes] Count - PTC127 (0x40DC; RC)

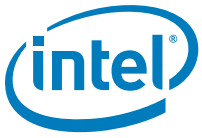
This register counts the number of packets transmitted that are 65-127 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*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.

7.18.50 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.



7.18.51 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.

7.18.52 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.

7.18.53 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, the I210-CS/CL 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.

7.18.54 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.

7.18.55 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.

7.18.56 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.

7.18.57 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.

7.18.58 Host Good Packets Transmitted Count-HGPTC (0x4118; RC)

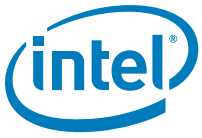
Field	Bit(s)	Initial Value	Description
HGPTC	31:0	0x0	Number of good packets transmitted by the host.

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. This register only increments if transmits are enabled (*TCTL.EN* is set).

7.18.59 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.



7.18.60 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.

7.18.61 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 - 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).

7.18.62 Host Good Octets Transmitted Count - HGOTCL (0x4130; RC)

Field	Bit(s)	Initial Value	Description
HGOTCL	31:0	0x0	Number of good octets transmitted by host - lower 4 bytes.

7.18.63 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.

7.18.64 Length Error Count - LENERRS (0x4138; RC)

Field	Bit(s)	Initial Value	Description
LENERRS	31:0	0x0	Length error count.

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.

7.18.65 SerDes/SGMII/KX Code Violation Packet Count - SCVPC (0x4228; RW)

This register contains the number of code violation packets received. Code violation is defined as an invalid received code in the middle of a packet.

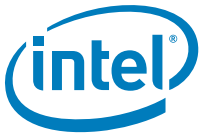
Field	Bit(s)	Initial Value	Description
CODEVIO	31:0	0x0	Code Violation Packet Count. At any point of time this field specifies number of unknown protocol packets received. Valid only in SGMII/SerDes/1000BASE-KX modes.

7.18.66 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.

7.19 Statistical Counters

The I210-CS/CL supports nine statistical counters per queue.



7.19.1 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 7.18.37). 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).

7.19.2 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.

7.19.3 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: VLAN tag is part of the byte count only if reported to the VM. For example, if the *DVMOLR.HIDE VLAN* bit is not set for this VM. 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.

7.19.4 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.

7.19.5 Per Queue Multicast Packets Received Count - PQMPRC (0x10038 + n*0x100 [n=0...3]; RO)

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.

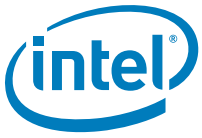
Note: Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
MPRC	31:0	0x0	Number of multicast packets received.

7.20 Wake Up Control Register Descriptions

7.20.1 Wake Up Control Register - WUC (0x5800; R/W)

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.



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 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 when the AUX_PWR = 0b bit is also reset on de-assertion of PE_RST_N and during D3 to D0 transition.
PME_Status (R/W1C)	2	0b	PME_Status. This bit is set when the I210-CS/CL 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 when the AUX_PWR = 0b bit is also reset on de-assertion of PE_RST_N and during D3 to D0 transition.
APMPME	3	0b[1]	Assert PME On APM Wakeup. If set to 1b, the I210-CS/CL 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 the I210-CS/CL 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 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 (refer to Section 7.21).
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 only.
Reserved	31:6	0x0	Reserved. Write 0x0, ignore on read.

1. Loaded from the Flash.

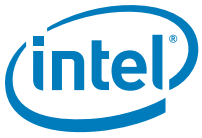
7.20.2 Wakeup Filter Control Register - WUFC (0x5808; R/W)

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.

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.

Field	Bit(s)	Initial Value	Description
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.
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= Reserved.
FLX1_ACT	25	0b	Flexible Filter 1 Action. 0b= WoL. 1b= Reserved.
FLX2_ACT	26	0b	Flexible Filter 2 Action. 0b= WoL. 1b= Reserved.
FLX3_ACT	27	0b	Flexible Filter 3 Action. 0b= WoL. 1b= Reserved.
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.



7.20.3 Wake Up Status Register - WUS (0x5810; R/W1C)

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.
Reserved	8	0b	Reserved.
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.

Field	Bit(s)	Initial Value	Description
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.

7.20.4 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.

7.20.5 Wake Up Packet Memory - WUPM (0x5A00 + 4*n [n=0...31]; RO)

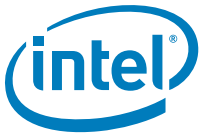
This register is read-only and it is used to store the first 128 bytes of the wake up packet for software retrieval after system wake up. It is not cleared by any reset.

Field	Bit(s)	Initial Value	Description
WUPD	31:0	X	Wakeup Packet Data.

7.20.6 Proxying Filter Control Register - PROXYFC (0x5F60; R/W)

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.

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.
BC	4	0b	Broadcast Proxy Enable.



Field	Bit(s)	Initial Value	Description
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.
Reserved	15	0b	Reserved.
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.

7.20.7 Proxying Status Register - PROXYS (0x5F64; R/W1C)

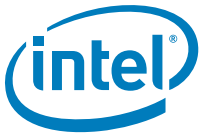
This register is used to record statistics about all proxying packets received. If a packet matches multiple criteria then multiple bits could be set. Writing a 1b to any bit clears that bit.

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.



7.20.8 Proxying Filter Control Extended Register - PROXYFCEX (0x5590; R/W)

This register is an extension to PROXYFC and is used to control and enable the routing to management (like 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.

7.20.9 Proxying Extended Status Register - PROXYEXS (0x5594; R/W1C)

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.

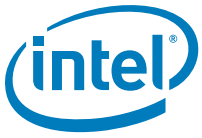
Note: If additional packets are received that matches one of the wake-up filters, after the original wake-up packet is received, the PROXYS register is updated with the matching filters accordingly.

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.

7.20.10 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.



Field	Bit(s)	Initial Value	Description
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.

7.20.11 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.

7.20.12 Range and Wake Port Filter Control - RWPFC (0x5584; RW)

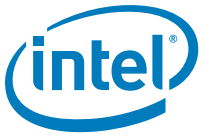
Field	Bit(s)	Initial Value	Description
RangeControl	1:0	00b	Range Port Filter Control. 00b = Port Filter disabled. 01b = UDP port. 10b = TCP port. 11b = TCP port and TCP flag SYN set; TCP flag RESET clear.
RangeAction	2	0b	The *Range Action* bit defines the action to take on a match to the range port 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.

7.20.13 Wake Flex UDP/TCP Ports Status - WFUTPS (0x5588, R/W1C)

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.
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.

7.20.14 Wake Control Status - WCS (0x558C, R/W1C)

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.



Field	Bit(s)	Initial Value	Description
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.

7.20.15 IP Address Valid - IPAV (0x5838; R/W)

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.

7.20.16 IPv4 Address Table - IP4AT (0x5840 + 8*n [n=0...3]; R/W)

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.

7.20.17 IPv6 Address Table - IP6AT (0x5880 + 4*n [n=0...3]; R/W)

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.

7.20.18 Flexible Host Filter Table Registers - FHFT (0x9000 + 4*n [n=0...255]; RW)

Each of the 8 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 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 7.20.18.1). When the I210-CS/CL 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.

Field	Bit(s)	Initial Value	Description
Bit Vector	31:0	X	The details of the bit vector are described in Table 7-24.

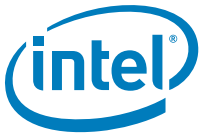


Table 7-24. FHFT Filter Description

31 0	31 8	7 0	31 0	31 0
Reserved	Reserved	Mask [7:0]	DW 1	Dword 0
Reserved	Reserved	Mask [15:8]	DW 3	Dword 2
Reserved	Reserved	Mask [23:16]	DW 5	Dword 4
Reserved	Reserved	Mask [31:24]	DW 7	Dword 6

....

31 8	7 0	31 8	7 0	31 0	31 0
Reserved	Reserved	Reserved	Mask [119:112]	DW 29	Dword 28
Queueing	Length	Reserved	Mask [127:120]	DW 31	Dword 30

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.

7.20.18.1 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 the I210-CS/CL 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.
Reserved	31:25	X	Reserved. Write 0x0, ignore on read.

7.20.18.2 Flex Filter 0 - Example

Field	Dword	Address	Bit(s)	Initial Value
Filter 0 DW0	0	0x9000	31:0	X
Filter 0 DW1	1	0x9004	31:0	X
Filter 0 Mask[7:0]	2	0x9008	7:0	X
Reserved	3	0x900C	31:0	X
Filter 0 DW2	4	0x9010	31:0	X
,Ä¶				
Filter 0 DW30	60	0x90F0	31:0	X
Filter 0 DW31	61	0x90F4	31:0	X
Filter 0 Mask[127:120]	62	0x90F8	7:0	X
Length	63	0x90FC	7:0	X
Filter 0 Queueing	63	0x90FC	31:8	X

7.20.19 Flexible Host Filter Table Extended Registers - FHFT_EXT (0x9A00 + 4*n [n=0...255]; 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. If enabled, the first 128 bytes of the received packet are compared against the non-masked bytes in the FHFT_EXT register. The layout and access rules of this table are the same as in FHFT.

Field	Bit(s)	Initial Value	Description
bit vector	31:0	X	The details of the bit vector are described in Table 7-24.

7.21 Host Interface Memory Registers Description

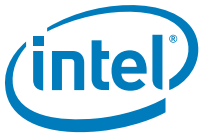
The software device driver communicates with the manageability block through CSR access.

7.21.1 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 ManagementDATA RAM: The internal DATA RAM is mapped to address 0x8800-0x8EFF. Writing to this address space goes directly to the RAM. This section can be limited by the internal firmware. The firmware reports to the host the maximum size allocated in the HIBSMAXOFF register. When this section is used for host interface commands it is usually limited to 512 bytes. When it is used to load firmware in systems (Section 3.3.3), the full 1792 bytes are available.
2. Control registers located at address 0x8F00.



7.21.1.1 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.
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.
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.

7.21.2 HOST Interface Control Register - HICR (0x8F00; RW)

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 internal DATA RAM. This bit should be cleared by the firmware after the command's processing completes.
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.
Reserved	6:4	0x0	Reserved.
FWR	7	0b	Firmware Reset. When set by the host, it indicates that the hardware needs to assert a firmware reset. This bit is meaningful only when in the non-secured mode.
Reserved	8	0b	Reserved.
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.

7.21.3 Host Interface Buffer Base Address - HIBBA (0x8F40; RW)

Notes:

1. This register is reset by a firmware reset.
2. 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.

7.21.4 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.

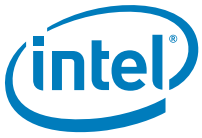
Field	Bit(s)	Initial Value	Description
MAXOFF	9:0	0x3FF	Maximum offset in the HIB for the CSR slave access. The 2 LSBs are always set to 11b.
Reserved	31:10	0x0	Reserved. Write 0x0, ignore on read.

7.22 Memory Error Registers Description

Main internal memories are protected by Error Correcting Code (ECC) or parity bits. The I210-CS/CL contains several registers that enable and report detection of internal memory errors. Description and usage of these registers can be found in Section 7.6.

7.22.1 Parity and ECC Error Indication- PEIND (0x1084; RC)

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.



Field	Bit(s)	Initial Value	Description
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.

7.22.2 Parity and ECC Indication Mask - PEINDM (0x1088; RW)

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.*
mng_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.

7.22.3 Packet Buffer ECC Status - PBECCSTS (0x245C; R/W)

Field	Bit(s)	Init.	Description
ecc_en	0	0x1	ECC Enable.
Reserved	1	0x0	Reserved Write 0, ignore on read.
pb_cor_err_sta (R/W1C)	2	0x0	DBU RAM correctable error indication. Bit is clean by write 1b.
Reserved	31:3	0x0	Reserved. Write 0x0, ignore on read.

7.22.4 PCIe Parity Control Register - PCIEERRCTL (0x5BA0; RW)

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

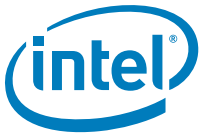
Field	Bit(s)	Initial Value	Description
Reserved	7	0b	Reserved.
ERR EN RX CDQ 1	8	1b	RX CDQ 1 Parity Check Enable.
Reserved	9	0b	Reserved.
ERR EN RX CDQ 2	10	1b	RX CDQ 2 Parity Check Enable.
Reserved	11	0b	Reserved.
ERR EN RX CDQ 3	12	1b	RX CDQ 3 Parity Check Enable.
Reserved	31:13	0x0	Reserved.

1. Bit loaded from Flash.

7.22.5 PCIe Parity Status Register - PCIEERRSTS (0x5BA8; R/W1C)

Register logs uncorrectable parity errors detected in PCIe logic.

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. 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.RST* and re-initializing the port (refer to Section 6.6.1.1). ***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. 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.RST* and re-initializing the port (refer to Section 6.6.1.1). ***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. 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.RST* and re-initializing the port (refer to Section 6.6.1.1). ***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. 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.RST* and re-initializing the port (refer to Section 6.6.1.1). ***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.



7.22.6 PCIe ECC Control Register - PCIEECCCTL (0x5BA4; RW)

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.
Reserved	13	0b	Reserved.
ERR EN RETRY BUF	14	1b	Tx Retry Buffer ECC Check Enable.
Reserved	31:15	0x0	Reserved. Write 0x0, Ignore on read.

7.22.7 PCIe ECC Status Register - PCIEECCSTS (0x5BAC; R/W1C)

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
ECC ERR RETRY BUF	5	0b	TX Retry Buffer ECC Correctable Error
Reserved	31:6	0x0	Reserved Write 0, ignore on read

7.22.8 PCIe ACL0 and ACL1 Register - PCIACL01 (0x5B7C; RO to Host)

Note: Reset by PCIe reset.

Field	Bit(s)	Initial Value	Description
ACL0	15:0	0	One of the four ACLs.
ACL1	31:16	0	One of the four ACLs.

7.22.9 PCIe ACL2 and ACL3 Register - PCIACL23 (0x5B80; RO to Host)

Note: Reset by PCIe reset.

Field	Bit(s)	Initial Value	Description
ACL2	15:0	0	One of the four ACLs.
ACL3	31:16	0	One of the four ACLs.

7.22.10 LAN Port Parity Error Control Register - LANPERRCTL (0x5F54; RW)

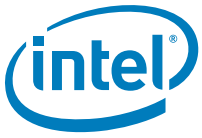
Field	Bit(s)	Initial Value	Description
Reserved	8:0	0x0	Reserved. Write 0x0, ignore on read.
retx_buf_en	9	1b	Enable retx_buf parity error indication When set to 1b, enables the RETX buffer (re-transmit buffer) parity error detection and indication.
Reserved	31:10	0x0	Reserved. Write 0x0, ignore on read.

7.22.11 LAN Port Parity Error Status Register - LANPERRSTS (0x5F58; R/W1C)

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.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.

7.23 Power Management Register Description

The following registers are used to control various power saving features.



7.23.1 DMA Coalescing Control Register - DMACR (0x2508; R/W)

Field	Bit(s)	Initial Value	Description
DMACWT	13:0	0x20	DMA Coalescing Watchdog Timer. When in DMA coalescing, the value in the *DMACR.DMACWT* counter sets the upper limit in 32.768 µs units between receive packet arrival as well as the request to transmit or issue an interrupt cause to move out of DMA coalescing. ***Note:*** If the value is 0x0, a condition to move out of DMA coalescing is a result of the watchdog timer expiration being disabled.
Reserved	14	0b	Reserved.
DC_BMC2OSW_EN	15	1b	DMA Coalescing MC-to-OS Watchdog Enable. When set to 1b, MC-to-OS traffic activate the DMA coalescing watchdog timer (*DMACR.DMACWT*). ***Note:*** If the DMA coalescing watchdog timer is disabled and this bit is set 1b, any MC-to-OS traffic causes a move out of the DMA coalescing state.
DMACTHR	23:16	0x0	DMA Coalescing Receive Threshold. This value defines the DMA coalescing receive threshold in 1 KB units. When the amount of data in the internal receive buffer exceeds the *DMACTHR* value, DMA coalescing is stopped and PCIe moves to the L0 state. ***Notes:*** 1. This value should be lower than the *FCRTC.RTH_Coal* threshold value to avoid generating needless flow control packets when in DMA coalescing operating mode and flow control is enabled. 2. The receive threshold size should be smaller than the internal receive buffer area reported in the *RXPBSIZE.RXPbsize* field. 3. If the value is 0x0, condition to move out of DMA coalescing as a result of passing DMA coalescing receive threshold is disabled. 4. The value programmed should be greater than maximum packet size.
Reserved	24	0b	Reserved. Write 0b, ignore on read.
EXIT_DC (SC)	25	0b	Exit DMA Coalescing. Software can initiate a one time move out of the DMA coalescing state by setting this bit to 1b.
Reserved	27:26	00b	Reserved.
Reserved	29:28	11b	Reserved. Write 11b, ignore on read.
Reserved	30	0b	Reserved. Write 0b, ignore on read.
DMAC_EN	31	0b	DMA Coalescing Enable. 0b = Disable DMA Coalescing. 1b = Enable DMA Coalescing.

7.23.2 DMA Coalescing Transmit Threshold - DMCTXTH (0x3550;RW)

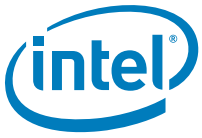
Field	Bit(s)	Initial Value	Description
DMCTTHR	11:0	0xE4	DMA Coalescing Transmit Threshold. This value defines the DMA coalescing transmit threshold in 64 byte units. When the amount of empty space in the internal transmit buffer exceeds the DMCTTHR value and additional transmit data is available in main memory, DMA coalescing is stopped and PCIe moves to an L0 state. ***Notes:*** 1. If this value is 0x0 or smaller than the maximum transmit packet size, as defined in the *DTXMXPKTSZ.MAX_TPKT_SIZE* field, a condition to move out of DMA coalescing due to the passing of the DMA coalescing transmit threshold level is disabled. 2. The transmit threshold size should be smaller than the internal transmit buffer area reported in the *TXPBSIZE* field.
Reserved	31:12	0b	Reserved. Write 0x0, ignore on read.

7.23.3 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.
Reserved	31:20	0b	Reserved. Write 0x0, ignore on read.

7.23.4 DMA Coalescing Time to Lx Request - DMCTLX (0x2514;RW)

Field	Bit(s)	Initial Value	Description
Reserved	11:0	0x20	Reserved. Write 0x20, ignore on read.
Reserved	12	0b	Reserved.
Reserved	30:13		Reserved. Write 0x0, ignore on read.
DCFLUSH_DIS	31	0b	Disable DMA Coalescing Flush. When this bit is set, the flush of pending interrupts and pending descriptor write-back operations before entry into DMA Coalescing is disabled.



7.23.5 DMA Coalescing Current Rx Count - DMCCNT (0x5DD4;RO)

Field	Bit(s)	Initial Value	Description
CCOUNT	24:0	0x0	DMA Coalescing Receive Traffic Current Count. Represents the count of receive traffic in the current time interval in units of 64-byte segments. ***Note:*** Counter does not wrap around.
RSVD	31:25	0x0	Reserved. Write 0x0, ignore on read.

7.23.6 Flow Control Receive Threshold Coalescing - FCRTC (0x2170; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0x0	Reserved. Write 0x0 ignore on read.
RTH_Coal	17:4	0x0	Flow control receive threshold high watermark value used to generate a XOFF flow control packet when executing DMA coalescing, internal transmit FIFO is empty and transmit flow control is enabled (*CTRL.TFCE* = 1b). When previous conditions exist, a XOFF packet is sent if the occupied space in the Rx packet buffer is more or equal to this watermark. This field is in 16 bytes granularity. Refer to Section 3.6.5.3.1to calculate the *FCRTC.RTH_Coal* value. ***Notes:*** 1. To avoid sending XOFF flow control packets needlessly when executing DMA coalescing and the internal transmit buffer is empty, the value should be higher than the threshold defined in the *DMACR.DMACTHR* field. Maximum threshold value can be up to *FCRTH0.RTH* + maximum allowable packet size * 1.25. 2. *RTH_Coal* threshold value is used as a watermark for sending flow control packets when DMA coalescing is enabled and the internal transmit buffer is empty. 3. The value programmed should be greater than the maximum packet size.
Reserved	31:18	0x0	Reserved Write 0 ignore on read.

7.23.7 DMA Coalescing Clock Control Time Counter - DMACTC (0x5DC8; RO)

This register keeps track of the number of time units elapsed since the end of last time interval.

Field	Bit(s)	Initial Value	Description
COUNT	9:0	0x0	SC Time Counter. The counter for the number of time units elapsed since the end of the last time interval.
RSVD	31:10	0x0	Reserved.

7.24 Diagnostic Registers Description

7.24.1 PCIe Misc. Register - PCIEMISC (0x5BB8; RW)

Note: Reset by PCIe power good reset.

Field	Bit(s)	Initial Value	Description
Reserved	8:0	0x8A	Reserved Ignore on read, write 0x8A.
DMA Idle Indication	9	0b[1]	Indication For DMA Idle This bit indicates when DMA is considered idle (either when the DMA is idle or when PCIe Link is idle). 0b = DMA is considered idle when there is no Rx or Tx. 1b = DMA is considered idle when there is no Rx or Tx AND when there are no TLPs indicating that CPU is active detected on the PCIe link (such as the host executes CSR or Configuration register read or write operation). ***Note:*** The bit must be set to 1b each time programming the Flash via CSR accesses.
Reserved	31:10	0x122	Reserved Ignore on read, write 122.

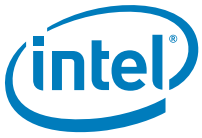
1. Value loaded from Flash.

Pulses shorter than the filter width are ignored.

7.24.2 PHY Registers

7.24.2.1 Fiber Control Register - Page 26, Register 16

Bits	Field	Mode	HW Rst	SW Rst	Description
15	Fiber Reset	R/W	0x0	SC	Fiber Software Reset. Affects page 1. Writing a 0x1 to this bit causes the PHY state machines to be reset. When the reset operation completes, this bit is cleared to 0x0 automatically. The reset occurs immediately. 1b = PHY reset. 0b = Normal operation.
14	Loopback	R/W		Retain	The latest event that occurs between the register write and pin control determines the loopback. When loopback is activated, the transmitter data presented on TXD of the internal bus is looped back to RXD of the internal bus. 1b = Enable loopback. 0b = Disable loopback.



13:12	Reserved	R/W	0x0	SC	Always 0x0.
11:10	Power Down	R/W		0x0	The latest event that occurs between the register write and pin control determines the register value. 00b = Total power up. 01b = Power down everything except wire activity detection circuit. 1xb = Total power down.
9:0	Reserved	RO	Always 0x0	Always 0x0	Always 0x0.

7.24.2.2 Fiber Control Register - Page 26, Register 21

Bits	Field	Mode	HW Rst	SW Rst	Description
15:0	Receive Error Count	RO, LH	0x0	Retain	Counter reaches its maximum count at 0xFFFF and does not roll over. Both false carrier and symbol errors are reported.

7.24.2.3 PRBS Control - Page 26, Register 23

Bits	Field	Mode	HW Rst	SW Rst	Description
15:8	Reserved	R/W	0x0	Retain	Set to 0x0.
7	Invert Checker Polarity	R/W	0x0	Retain	0 = Normal. 1 = Invert.
6	Invert Generator Polarity	R/W	0x0	Retain	0 = Normal. 1 = Invert.
5	PRBS Lock	R/W	0x0	Retain	0 = Counter free runs. 1 = Do not start counting until PRBS locks first.
4	Clear Counter	R/W, SC	0x0	0x0	0 = Normal. 1 = Clear counter.
3:2	Pattern Select	R/W	00	Retain	00b = PRBS 7. 01b = PRBS 23. 10b = PRBS 31. 11b = Generate 1010101010... pattern.
1	PRBS Checker Enable	R/W	0x0	0x0	0b = Disable. 1b = Enable.
0	PRBS Generator Enable	R/W	0x0	0x0	0b = Disable. 1b = Enable.

7.24.2.4 PRBS Error Counter LSB- Page 26, Register 24

Bits	Field	Mode	HW Rst	SW Rst	Description
15:0	PRBS Error Count LSB	RO	0x0	Retain	A read to this register freezes register 25_26. Cleared only when register 23_26.4 is set to 0b.

7.24.2.5 PRBS Error Counter MSB- Page 26, Register 25

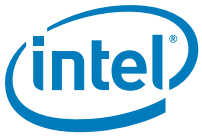
Bits	Field	Mode	HW Rst	SW Rst	Description
15:0	PRBS Error Count MSB	RO	0x0	Retain	This register does not update unless register 24_26 is read first. Cleared only when register 23_26.4 is set to 1b.

7.24.2.6 Fiber Specific Control Register 2 - Page 26, Register 26

Bits	Field	Mode	HW Rst	SW Rst	Description
15:3	Reserved	R/W	0x0	Update	Must be set to 0x0.
2:0	Output Amplitude	R/W	0x2	Retain	Differential voltage peak measured. The latest event that occurs between the register write and pin control determines the current output amplitude setting. 000b = 14 mV. 001b = 112 mV. 010b = 210 mV. 011b = 308 mV. 100b = 406 mV. 101b = 504 mV. 110b = 602 mV. 111b = 700 mV.

7.24.2.7 Polarity Control - Page 26, Register 27

Bits	Field	Mode	HW Rst	SW Rst	Description
15	Invert rxp/n Polarity	R/W		Retain	The latest event that occurs between the register write and pin control determines the polarity. 0b = Normal. 1b = Invert.
14	Invert txp/n Polarity	R/W		Retain	The latest event that occurs between the register write and pin control determines the polarity. 0b = Normal. 1b = Invert.
13:2	Reserved	R/W	0x0	Retain	Reserved for future use.
1:0	SQ Control Selection	R/W	0x	Retain	Squelch detector threshold control. 00b = 30 mV. 01b = 60 mV. 10b = 90 mV. 11b = 120 mV.



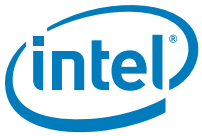
7.24.2.8 SerDes TX FIFO Control and Status - Page 26, Register 28

Bits	Field	Mode	HW Rst	SW Rst	Description
15:14	SerDes Transmit FIFO Depth	R/W	0x0	Retain	00b = Read/write pointers are offset by two cycles. 01b = Read/write pointers are offset by three cycles. 1xb = Reserved.
13:2	Reserved	R/W	0x0	Retain	0x0.
1	FIFO Full	RC	0b	Retain	1b = FIFO full, clear upon read.
0	FIFO Empty	RC	0b	Retain	1b = FIFO empty, clear upon read.

7.24.2.9 Voltage Regulator Control - Page 26, Register 30

Bits	Field	Mode	HW Rst	SW Rst	Description
15:7	Reserved	R/W	0x0	Retain	0x0.
6:3	scr09 Output Voltage Select	R/W		Retain	The latest event that occurs between the register write and pin control determines the current output amplitude setting. 0000b = 0.70V 0001b = 0.725V 0010b = 0.75V 0011b = 0.775V 0100b = 0.80V 0101b = 0.825V 0110b = 0.85V 0111b = 0.875V 1000b = 0.90V (default) 1001b = 0.925V (not recommended) 1010b = 0.95V (not recommended) 1011b = 0.975V (not recommended) 1100b = 1.00V (not recommended) 1101b = 1.025V (not recommended) 1110b = 1.05V (not recommended) 1111b = 1.075V (not recommended)
2:0	scr15 Output Voltage Select	R/W		Retain	The latest event that occurs between the register write and pin control determines the current output amplitude setting. 000b = 1.35V. 001b = 1.40V. 010b = 1.45V. 011b = 1.50V, default. 100b to 111b= Reserved.

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



8.0 PCIe Programming Interface

8.1 PCIe* Compatibility

PCIe is completely compatible with existing deployed PCI software. To achieve this, PCIe hardware implementations conform to the following requirements:

- All devices required to be supported by deployed PCI software must be enumerable as part of a tree through PCI device enumeration mechanisms.
- Devices in their default operating state must conform to PCI ordering and cache coherency rules from a software viewpoint.
- PCIe devices must conform to PCI power management specifications and must not require any register programming for PCI-compatible power management beyond those available through PCI power management capabilities registers. Power management is expected to conform to a standard PCI power management by existing PCI bus drivers.
- PCIe devices implement all registers required by the PCI specification as well as the power management registers and capability pointers specified by the PCI power management specification. In addition, PCIe defines a PCIe capability pointer to indicate support for PCIe extensions and associated capabilities.

The function contain the following regions of the PCI configuration space:

- Mandatory PCI configuration registers
- Power management capabilities
- MSI and MSI-X capabilities
- PCIe extended capabilities

8.2 PCIe Register Map

8.2.1 Register Attributes

Configuration registers are assigned one of the attributes described in the following table.

Table 8-1. 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.

Table 8-1. Configuration Registers (Continued)

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 strapping or serial Flash. 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 8-2. Refer to a detailed description for registers loaded from the Flash at initialization time. Note that initialization values of the configuration registers are marked in parenthesis.

8.2.2 PCIe Configuration Space Summary

Table 8-2. 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	

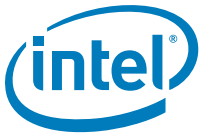


Table 8-2. PCIe Configuration Registers Map

Section	Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
MSI capability	0x50	Message Control (0x0080)		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 (0x00090)		Next Pointer (0xA0)	Capability ID (0x11)
	0x74	Table Offset			
	0x78	PBA offset			
CSR Access Registers	0x98	IOADDR			
	0x9C	IODATA			
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. (0x140)	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			
Serial ID capability	0x140	Next Capability Ptr. (0x1A0)	Version (0x1)	Serial ID Capability ID (0x0003)	
	0x144	Serial Number Register (Lower Dword)			
	0x148	Serial Number Register (Upper Dword)			

Table 8-2. PCIe Configuration Registers Map

Section	Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
TPH Requester capability	0x1A0	Next Capability Ptr. (0x1C0)	Version (0x1)	TPH Capability ID (0x17)	
	0x1A4	TPH Requester Capability Register			
	0x1A8	TPH Requester Control Register			
	0x1AC: 0x1B8	TPH Steering Table			

A description of the registers is provided in the following sections.

8.3 Mandatory PCI Configuration Registers

8.3.1 Vendor ID (0x0; RO)

This value can be loaded automatically from Flash address 0x0E at power up or reset. A value of 0x8086 is the default for this field at power up if the Flash does not respond or is not programmed.

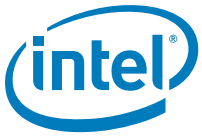
Note: To avoid a system hang situation, if a value of 0xFFFF is read from the Flash, the value of the Vendor ID field defaults back to 0x8086.

8.3.2 Device ID (0x2; RO)

This is a read-only register. This field identifies individual I210-CS/CL functions. It can be auto-loaded from the Flash during initialization with a different value. The following table lists the possible values according to the SKU and functionality.

PCI Function	Default Value	Flash Address	Description
LAN 0	0x1533 for I210-CS/CL SKUs with a programmed Flash	0x0D	0x1531 - I210-CS/CL with a blank Flash (tools only, not for driver) 0x1533 - I210-CS/CL 10/100/1000 Mb/s Ethernet controller, copper only[2] 0x1534 - Reserved 0x1536 - I210-CS/CL 10/100/1000 Mb/s Ethernet controller, Fiber [3] 0x1537 - I210-CS/CL 10/100/1000 Mb/s Ethernet controller, 1000BASE-KX/BX backplane [4] 0x1538 - I210-CS/CL 10/100/1000 Mb/s Ethernet controller, External SGMII PHY [5]

2. CTRL_EXT.Link_Mode field value 00b (10/100/1000 BASE-T internal PHY mode).
3. CTRL_EXT.Link_Mode field value 11b (SerDes).
4. CTRL_EXT.Link_Mode field value either 01b (1000BASE-KX) or 11b (SerDes - 1000BASE-BX). User option to enable Clause 37 Auto-negotiation.
5. CTRL_EXT.Link_Mode field value 10b (SGMII).



8.3.3 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 Flash 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.

8.3.4 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. The I210-CS/CL 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.

8.3.5 Revision (0x8; RO)

The default revision ID for the I210-CS/CL A1 stepping is 0x01 and 0x03 for A2 stepping. The value of the rev ID is a logic XOR between the default value and the value in Flash word 0x1E.

8.3.6 Class Code (0x9; RO)

The class code is a RO hard coded value that identifies the I210-CS/CL's functionality.

- 0x020000/0x010000 - Ethernet/SCSI Adapter[1]

8.3.7 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) Flash word and defines cache line size in Dwords. In systems, the value is 0x10.

8.3.8 Latency Timer (0xD; RO)

Not used. Hardwired to zero.

8.3.9 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.

8.3.10 BIST (0xF; RO)

BIST is not supported in the I210-CS/CL.

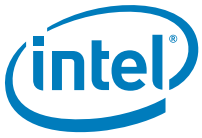
8.3.11 Base Address Registers (0x10...0x27; R/W)

The Base Address registers (BARs) are used to map the I210-CS/CL register space. The I210-CS/CL has a memory BAR, IO BAR and MSI-X BAR described in Table 8-3 below.

Table 8-3. 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 FLBARSize and CSRSize fields in the BARCTRL register. 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 Flash "*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.

1. Selected according to bit 11 in *Device Rev ID* Flash word.



8.3.11.1 32-bit LAN BARs Mode Mapping

This mapping is selected when bit 10 in the *Functions Control* Flash word is equal to 1b.

Table 8-4. Base Address Setting in 32bit BARs Mode (*BARCTRL.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
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)					

8.3.11.2 64-bit LAN BARs Mode Mapping

This mapping is selected when bit 10 in the *Functions Control* Flash word is equal to 0b.

Table 8-5. Base Address Setting in 64bit BARs Mode (*BARCTRL.BAR32* = 0b)

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
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)					

8.3.11.3 Base Address Register Fields

All base address registers have the following fields.

Table 8-6. Base Address Registers' Fields

Field	Bits	R/W	Description
Mem / IO Space Indication	0	RO	0b = Indicates memory space. 1b = Indicates I/O.
Memory Type	2:1	RO	00b = 32-bit BAR (BAR32 in the Flash equals 1b) 10b = 64-bit BAR (BAR32 in the Flash 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 Flash. This bit should be set only on systems that do not generate prefetchable cycles.

Table 8-6. Base Address Registers' Fields

Field	Bits	R/W	Description	
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 BARCTRL.FLBARSize and BARCTRL.CSRSize fields.	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

8.3.12 CardBus CIS (0x28; RO)

Not used. Hardwired to zero.

8.3.13 Subsystem Vendor ID (0x2C; RO)

This value can be loaded automatically from Flash address 0x0C at power up or reset. A value of 0x8086 is the default for this field at power up if the Flash does not respond or is not programmed.

8.3.14 Subsystem ID (0x2E; RO)

This value can be loaded automatically from Flash address 0x0B at power up with a default value of 0x0000.

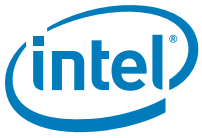
8.3.15 Expansion ROM Base Address (0x30; RW)

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* Flash 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	10:1	RO	0b	Always read as 0b. Writes are ignored.
Address	31:11	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* Flash 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).

8.3.16 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). The I210-CS/CL 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.



8.3.17 Interrupt Line (0x3C; RW)

Read/write register programmed by software to indicate which of the system interrupt request lines this I210's interrupt pin is bound to. See the PCIe definition for more details.

8.3.18 Interrupt Pin (0x3D; RO)

Read only register. Always report INTA#.

8.3.19 Max_Lat/Min_Gnt (0x3E; RO)

Not used. Hardwired to zero.

8.4 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 the I210-CS/CL.

Table 8-7. 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

8.4.1 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 Flash at initialization time. Behavior of some fields in this section depend on the *Power Management* bit in Flash 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	

8.4.1.1 Capability ID (0x40; RO)

This field equals 0x01 indicating the linked list item as being the PCI Power Management registers.

8.4.1.2 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.

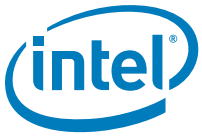
8.4.1.3 Power Management Capabilities - PMC (0x42; RO)

This field describes the I210-CS/CL'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	01001b See value in description column	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(12) X XX1Xb - PME# can be asserted from D1 bit(13) X X1XXb - PME# can be asserted from D2 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) Flash word. **Condition** — **Functionality** — **Value** PM Dis in Flash — No PME at all states — 00000b PM Ena & NoAux Pwr — PME at D0 and D3hot — 01001b PM Ena & Aux Pwr — PME at D0, D3hot and D3cold — 11001b Note: Aux Pwr is considered available if AUX_PWR pin is connected to 3.3V and *D3COLD_WAKEUP_ADVEN* Flash bit is set to 1b.
10	0b	RO	D2_Support The I210-CS/CL does not support D2 state.
9	0b	RO	D1_Support The I210-CS/CL does not support D1 state.
8:6	000b	RO	AUX Current – Required current defined in the Data Register.
5	1b	RO	DSI The I210-CS/CL 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 The I210-CS/CL complies with the PCI PM specification, revision 1.2.

8.4.1.4 Power Management Control / Status Register - PMCSR (0x44; R/W)

This register is used to control and monitor power management events in the I210-CS/CL. 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) Flash word and the *Data_Select* field is set to 0, 3, 4, 7, (or 8). 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) Flash word.
8	0b (at power up)	R/WS	PME_En If power management is enabled in the Flash, writing a 1b to this register enables wake up. If power management is disabled in the Flash, writing a 1b to this bit has no effect and does not set the bit to 1b.
7:4	000000b	RO	Reserved
3	1b[1]	RO	No_Soft_Reset No_Soft_Reset - When set (“1”), this bit indicates that when the I210-CS/CL 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”), the I210-CS/CL 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 Flash)

1. Loaded from Flash (See Section 6.2.17).

8.4.1.5 Bridge Support Extensions - PMCSR_BSE (0x46; RO)

This register is not implemented in the I210-CS/CL. Values are set to 0x00.

8.4.1.6 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 Flash if power management is enabled in the Flash or with a default value of 0x00. The values for the I210-CS/CL are read from Flash word 0x22.

Function	D0 (Consume/ Dissipate)	D3 (Consume/ Dissipate)	Common
PMCSR.Data Select	0x0 / 0x4	0x3 / 0x7	0x8
Function 0	Flash addr 0x22	Flash addr 0x22	Flash addr 0x22

For other *Data_Select* values, the Data register output is reserved (0x0).

8.4.2 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			
0x5C	Reserved		Message Data	
0x60	Mask bits			
0x64	Pending bits			

8.4.2.1 Capability ID (0x50; RO)

This field equals 0x05 indicating the linked list item as being the MSI registers.

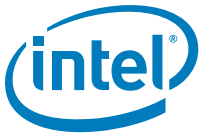
8.4.2.2 Next Pointer (0x51; RO)

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.

8.4.2.3 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, the I210-CS/CL generates an MSI for interrupt assertion instead of INTx signaling.
3:1	000b	RO	Multiple Message Capable The I210-CS/CL indicates a single requested message.
6:4	000b	RO	Multiple Message Enable The I210-CS/CL returns 000b to indicate that it supports a single message.



Bits	Default	R/W	Description
7	1b	RO	64-bit capable A value of 1b indicates that the I210-CS/CL is capable of generating 64-bit message addresses.
8	1b[1]	RO	MSI per-vector masking. A value of 1b indicates that the I210-CS/CL is capable of per-vector masking. This field is loaded from the *MSI-X Configuration* (Offset 0x16) Flash word.
15:9	0b	RO	Reserved Write 0 ignore on read.

1. Default value is read from the Flash.

8.4.2.4 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.

8.4.2.5 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.

8.4.2.6 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.

8.4.2.7 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 the I210-CS/CL 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, the I210-CS/CL is prohibited from sending MSI messages.
31:1	000b	RO	Reserved

8.4.2.8 Pending Bits (0x64; R/W)

Bits	Default	R/W	Description
0	0b	RO	If set, the I210-CS/CL has a pending MSI message.
31:1	000b	RO	Reserved

8.4.3 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 7.9, 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
```

The I210-CS/CL 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. The I210-CS/CL 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.

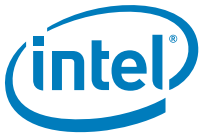


Table 8-8. MSI-X Capability Structure

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x70	Message Control (0x00090)		Next Pointer (0xA0)	Capability ID (0x11)
0x74	Table Offset			
0x78	PBA offset			

8.4.3.1 Capability ID (0x70; RO)

This field equals 0x11 indicating the linked list item as being the MSI-X registers.

8.4.3.2 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.

8.4.3.3 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. The I210-CS/CL supports 5 MSI-X vectors. This field is loaded from the *MSI-X Configuration* (Offset 0x16) Flash 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 Flash.

8.4.3.4 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. BIR values: 0...5 correspond to BARs 0x10…0x 24 respectively. A BIR value of 3 indicates that the table is mapped in BAR 3 (address 0x1C). When *BARCTRL.BAR32* equals 0b (64 bit MMIO mapping) the table BIR equals 0x4. When *BARCTRL.BAR32* equals 1b (32 bit MMIO mapping) the table BIR equals 0x3.

8.4.3.5 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. BIR values: 0...5 correspond to BARs 0x10…0x 24 respectively. A BIR value of 3 indicates that the table is mapped in BAR 3 (address 0x1C). When *BARCTRL.BAR32* equals 0b (64 bit MMIO mapping) the table BIR equals 0x4. When *BARCTRL.BAR32* equals 1b (32 bit MMIO mapping) the table BIR equals 0x3.

8.4.4 CSR Access Via Configuration Address Space

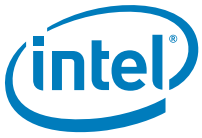
8.4.4.1 IOADDR Register (0x98; R/W)

This is a read/write register. Register is cleared at Power-up or PCIe reset.

Note: When function is in D3 state Software should not attempt to access CSRs via the *IOADDR* and *IODATA* registers.

Bit(s)	R/W	Initial Value	Description
30:0	R/W[1]	0x0	Internal Register or Internal Memory location Address. 0x00000-0x1FFFF – Internal Registers and Memories 0x20000-0x7FFFFFFF – Undefined
31	R/W	0b	Configuration IO Access Enable. 0b - CSR configuration read or write disabled. 1b - CSR Configuration read or write enabled When bit is set accesses to the IODATA register actually generate transactions to the device. Otherwise, accesses to the IODATA register are don't-cares (write are discarded silently, reads return arbitrary results).

1. In the event that the *CSR_conf_en* bit in the *PCIe Init Configuration 2* Flash word is cleared, accesses to the *IOADDR* register via configuration address space is ignored and has no effect on the register and the CSRs referenced by the *IOADDR* register.



8.4.4.2 IODATA Register (0x9C; R/W)

This is a read/write register. Register is cleared at Power-up or PCIe reset.

Bit(s)	R/W	Initial Value	Description
31:0	R/W[1]	0x0	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 are read/write-able.

1. In the event that the *CSR_conf_en* bit in the *PCIe Init Configuration 2* Flash word is cleared, access to the *IODATA* register via configuration address space is ignored and has no effect on the register and the CSRs referenced by the *IOADDR* register.

8.4.4.3 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.

8.4.5 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.

The I210-CS/CL implements the PCIe capability structure for endpoint devices as follows:

8.4.5.1 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.

8.4.5.2 PCIe CAP (0xA2; RO)

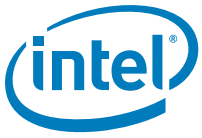
The PCIe capabilities register identifies the PCIe device type and associated capabilities. This is a read only register.

Bits	Default	R/W	Description
3:0	0010b	RO	Capability Version Indicates the PCIe capability structure version number. The I210-CS/CL supports both version 1 and version 2 as loaded from the PCIe *Capability Version* bit in the Flash.
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 The I210-CS/CL does not implement slot options therefore this field is hardwired to 0b.
13:9	00000b	RO	Interrupt Message Number The I210-CS/CL does not implement multiple MSI interrupts, therefore this field is hardwired to 0x0.
15:14	00b	RO	Reserved

8.4.5.3 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 the I210-CS/CL can support for TLPs. It is loaded from the Flash's *PCIe Init Configuration 3* word, 0x1A (with a default value of 512 bytes.
4:3	RO	00b	Phantom Function Supported Not supported by the I210-CS/CL.
5	RO	0b	Extended Tag Field Supported Max supported size of the *Tag* field. The I210-CS/CL supported 5-bit *Tag* field.
8:6	RO	011b	Endpoint L0s Acceptable Latency This field indicates the acceptable latency that the I210-CS/CL can withstand due to the transition from the L0s state to the L0 state. value loaded from the Flash *PCIe Init Configuration 1* word, 0x18 (See Section 6.2.14).
11:9	RO	110b	Endpoint L1 Acceptable Latency This field indicates the acceptable latency that the I210-CS/CL can withstand due to the transition from the L1 state to the L0 state. value loaded from the Flash *PCIe L1 Exit latencies* word, 0x14 (See Section 6.2.11).
12	RO	0b	Attention Button Present Hardwired in the I210-CS/CL to 0b.
13	RO	0b	Attention Indicator Present Hardwired in the I210-CS/CL to 0b.
14	RO	0b	Power Indicator Present Hardwired in the I210-CS/CL to 0b.
15	RO	1b	Role-Based Error Reporting This bit, when set, indicates that the I210-CS/CL 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 the I210-CS/CL.
17:16	RO	000b	Reserved
25:18	RO	0x00	Slot Power Limit Value Hardwired in the I210-CS/CL to 0x00, as the I210-CS/CL consumes less than the 25 W allowed for its form factor.



Bits	R/W	Default	Description
27:26	RO	00b	Slot Power Limit Scale Hardwired in the I210-CS/CL to 0b, as the I210-CS/CL 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 Flash.

8.4.5.4 Device Control (0xA8; RW)

This register controls the PCIe specific parameters.

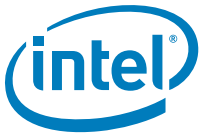
Bits	R/W	Default	Description
0	RW	0b	Correctable Error Reporting Enable Enable report of correctable errors.
1	RW	0b	Non-Fatal Error Reporting Enable Enable report of non fatal errors.
2	RW	0b	Fatal Error Reporting Enable Enable report of fatal errors.
3	RW	0b	Unsupported Request Reporting Enable Enable report of unsupported requests error.
4	RW	1b	Enable Relaxed Ordering If this bit is set, the I210-CS/CL 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 7.2.3.
7:5	RW	000b (128 bytes)	Max Payload Size This field sets maximum TLP payload size for the I210-CS/CL. As a receiver, the I210-CS/CL must handle TLPs as large as the set value. As a transmitter, the I210-CS/CL must not generate TLPs exceeding the set value. The max payload size supported in the I210-CS/CL 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 the I210-CS/CL.
9	R0	0b	Phantom Functions Enable Not implemented in the I210-CS/CL.
10	RWS	0b	Auxiliary Power PM Enable When set, enables the I210-CS/CL to draw AUX power independent of PME AUX power.

Bits	R/W	Default	Description
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.

8.4.5.5 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 the I210-CS/CL received an unsupported request.
4	RO	0b	Aux Power Detected If aux power is detected, this field is set to 1b. It is a strapping signal from the periphery. Reset on LAN_PWR_GOOD and GIO Power Good only.
5	RO	0b	Transactions Pending Indicates whether the I210-CS/CL has any transaction pending.
15:6	RO	0x00	Reserved



8.4.5.6 Link Capabilities Register (0xAC; RO)

This register identifies PCIe link specific capabilities. This is a read only register

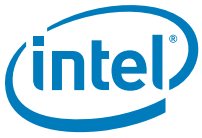
Bits	Rd/Wr	Default	Description
3:0	RO	0010b	Max Link Speed This field indicates the supported Link speed(s) of the associated link port. Defined encodings are: 0001b = 2.5 Gb/s Link speed supported. 0010b = Not supported (5 Gb/s and 2.5 Gb/s Link speeds)
9:4	RO	0x01	Max Link Width Indicates the maximum link width. The I210-CS/CL 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 the I210-CS/CL PCI Express Link. Defined encodings are: 00b = No ASPM Support. 01b = L0s Supported. 10b = L1 Supported. 11b = L0s and L1 Supported.
14:12	RO	Usage depended. See default values in Section 6.2.14.	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 Flash word, 0x18 (See Section 6.2.14).
17:15	RO	Usage depended. See default values in Section 6.2.11.	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* Flash word, 0x14 (See Section 6.2.11).
18	RO	0b	Clock Power Management Status Not supported in the I210-CS/CL. RO as zero.
19	RO	0b	Surprise Down Error Reporting Capable Status Not supported in the I210-CS/CL. RO as zero
20	RO	0b	Data Link Layer Link Active Reporting Capable Status Not supported in the I210-CS/CL. RO as zero.

Bits	Rd/Wr	Default	Description
21	RO	0b	Link Bandwidth Notification Capability Status Not supported in the I210-CS/CL. 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.

8.4.5.7 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 the I210-CS/CL 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 the I210-CS/CL 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.
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 the I210-CS/CL. RO as zero.
9	RO	0b	Hardware Autonomous Width Disable Not supported in the I210-CS/CL. RO as zero.



Bits	R/W	Default	Description
10	RO	0b	Link Bandwidth Management Interrupt Enable Not supported in the I210-CS/CL. RO as zero.
11	RO	0b	Link Autonomous Bandwidth Interrupt Enable Not supported in the I210-CS/CL. RO as zero.
15:12	RO	0000b	Reserved

8.4.5.8 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	0001b	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 = Not supported (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 the I210-CS/CL 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 the I210-CS/CL uses the physical reference clock that the platform provides on the connector. This bit must be cleared if the I210-CS/CL 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) Flash word.
13	RO	0b	Data Link Layer Link Active Not supported in the I210-CS/CL. RO as zero.
14	RO	0b	Link Bandwidth Management Status Not supported in the I210-CS/CL. RO as zero.
15	RO	0b	Reserved

8.4.5.9 Reserved (0xB4-0xC0; RO)

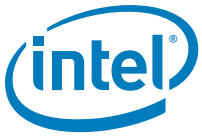
Unimplemented reserved registers not relevant to PCIe endpoint.

The following registers are supported only if the capability version is two and above.

8.4.5.10 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 the I210-CS/CL 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 the I210-CS/CL 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 the I210-CS/CL.
7	RO	0b	32-bit AtomicOp Completer Supported – not supported in the I210-CS/CL.
8	RO	0b	64-bit AtomicOp Completer Supported – not supported in the I210-CS/CL.
9	RO	0b	128-bit CAS Completer Supported – not supported in the I210-CS/CL.
10	RO	0b	No RO-enabled PR-PR Passing – not supported in the I210-CS/CL.
11	RO	1b	Reserved.
13:12	RO	00b	TPH Completer supported - the I210-CS/CL does not use the hints as a completer
17:14	RO	0x0	Reserved
19:18	RO	00b	Reserved
31:20	RO	0x0	Reserved



8.4.5.11 Device Control 2 (0xC8; RW)

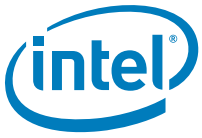
This register controls PCIe specific parameters.

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 the I210-CS/CL 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 the I210-CS/CL is 50 µs to 100 µs. • 0010b = Allowable range is 1 ms to 10 ms. Actual completion timeout range supported in the I210-CS/CL 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 the I210-CS/CL is 16 ms to 32 ms. • 0110b = Allowable range is 65 ms to 210 ms. Actual completion timeout range supported in the I210-CS/CL 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 the I210-CS/CL is 260 ms to 520 ms. • 1010b = Allowable range is 1 s to 3.5 s. Actual completion timeout range supported in the I210-CS/CL 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 the I210-CS/CL is 4 s to 8 s. • 1110b = Allowable range is 17 s to 64 s. Actual completion timeout range supported in the I210-CS/CL 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 the I210-CS/CL.
7	RO	0b	AtomicOp Egress Blocking - not supported in the I210-CS/CL.
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	Reserved.
12:11	RO	0x0	Reserved.
14:13	RW/ RO	00b	Reserved.
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 the I210-CS/CL is installed. This ensures that the I210-CS/CL 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.

8.4.5.12 Link Control 2 (0xD0; RW)

Bits	R/W	Default	Description
3:0	RWS	0001b	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 = Not supported (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 the I210-CS/CL (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	RO	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. Bit is Hard wired to 0b.
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.
11	RWS	0b	Compliance SOS When set to 1b, the LTSSM is required to send SOS periodically in between the (modified) compliance patterns.
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. Encodings: 1b -3.5 dB 0b -6 dB When the Link is operating at 2.5 GT/s, the setting of this bit has no effect.
15:13	RO	0x0	Reserved



8.4.5.13 Link Status 2 (0xD2; RW)

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

8.5 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. The I210-CS/CL decodes an additional 4-bits (bits 27:24) to provide the additional configuration space as shown in Table 8-9. 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 8-9. 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.

The I210-CS/CL supports the following PCIe extended capabilities.

Table 8-10. PCIe Extended Capability Structure

Capability	Offset	Next Header[1]
Advanced Error Reporting	0x100	0x140
Serial Number	0x140	0x1A0
TLP processing hints	0x1A0	0x1C0
Latency Tolerance Requirement Reporting	0x1C0	0x000

1. Some of the capabilities might be skipped if disabled via Flash.

8.5.1 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.

8.5.1.1 PCIe CAP ID (0x100; RO)

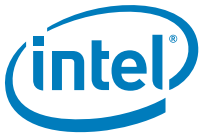
Bit Location	Attribute	Default Value	Description
15:0	RO	0x0001	Extended Capability ID PCIe extended capability ID indicating AER capability.
19:16	RO	0x2[1]	AER Capability Version PCIe AER extended capability version number.
31:20	RO	0x140	Next Capability Pointer Next PCIe extended capability pointer. A value of 0x140 points to the serial ID capability.

1. Loaded from Flash (See Section 6.2.19).

8.5.1.2 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 the I210-CS/CL.
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



Bit Location	Attribute	Default Value	Description
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 the I210-CS/CL.
22	RO	0b	Uncorrectable Internal Error Status (Optional) Not supported in the I210-CS/CL.
23	RO	0b	MC Blocked TLP Status (Optional) Not supported in the I210-CS/CL.
24	RO	0b	AtomicOps Egress Blocked Status (Optional) Not supported in the I210-CS/CL.
25	RO	0b	TLP Prefix Blocked Error Status (Optional) Not supported in the I210-CS/CL.
31:26	RO	0x0	Reserved

8.5.1.3 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 the I210-CS/CL.
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
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 the I210-CS/CL.
22	RO	0b	Uncorrectable Internal Error Mask (Optional) Not supported in the I210-CS/CL.
23	RO	0b	MC Blocked TLP Mask (Optional) Not supported in the I210-CS/CL.

Bit Location	Attribute	Default Value	Description
24	RO	0b	AtomicOps Egress Blocked Mask (Optional) Not supported in the I210-CS/CL.
25	RO	0b	TLP Prefix Blocked Error Mask (Optional) Not supported in the I210-CS/CL.
31:26	RO	0x0	Reserved

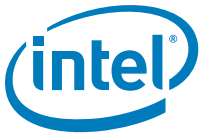
8.5.1.4 Uncorrectable Error Severity (0x10C; RWS)

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 the I210-CS/CL.
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 the I210-CS/CL.
22	RO	1b	Uncorrectable Internal Error Severity (Optional) Not supported in the I210-CS/CL.
23	RO	0b	MC Blocked TLP Severity (Optional) Not supported in the I210-CS/CL.
24	RO	0b	AtomicOps Egress Blocked Severity (Optional) Not supported in the I210-CS/CL.
25	RO	0b	TLP Prefix Blocked Error Severity (Optional) Not supported in the I210-CS/CL.
31:26	RO	0x0	Reserved

8.5.1.5 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 the I210-CS/CL.
15	RO	0b	Header Log Overflow Status (Optional) Not supported in the I210-CS/CL.
31:16	RO	0x0	Reserved

8.5.1.6 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.
14	RO	0b	Corrected Internal Error Mask (Optional) Not supported in the I210-CS/CL.
15	RO	0b	Header Log Overflow Mask (Optional) Not supported in the I210-CS/CL.
31:16	RO	0x0	Reserved

8.5.1.7 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 the I210-CS/CL is capable of generating ECRC. This bit is loaded from Flash 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 Flash 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

8.5.1.8 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.

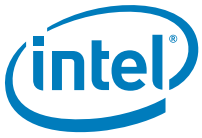
Bit Location	Attribute	Default Value	Description
127:0	ROS	0b	Header of the packet in error (TLP or DLLP).

8.5.2 Serial Number

The PCIe device serial number capability is an optional extended capability that can be implemented by any PCIe device. The device serial number is a read-only 64-bit value that is unique for a given PCIe device.

Note: The I210-CS/CL does not support this capability in a configuration.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x140	Next Capability Ptr. 0x1A0[1]	Version (0x1)	Serial ID Capability ID (0x0003)	
0x144	Serial Number Register (Lower Dword)			
0x148	Serial Number Register (Upper Dword)			



1. This value might change if the TPH capability is disabled. In this case the next header is the next enabled feature.

8.5.2.1 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	0x1A0	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.

8.5.2.2 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 8-11 lists the allocation of register fields in the Serial Number register. Table 8-11 also lists the respective bit definitions.

Table 8-11. Serial Number Register

31:0
Serial Number Register (Lower Dword)
Serial Number Register (Upper word)
63:32

Serial number definition in the I210-CS/CL:

Table 8-12. 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 Flash into the RAL and RAH registers.

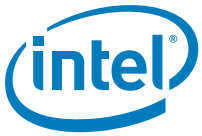
The translation from Flash words 0 to 2 to the serial number is as follows:

- Serial number ADDR+0 = Flash byte 5
- Serial number ADDR+1 = Flash byte 4
- Serial number ADDR+2 = Flash byte 3
- Serial number ADDR+3 and 4 = 0xFF 0xFF
- Serial number ADDR+5 = Flash byte 2
- Serial number ADDR+6 = Flash byte 1
- Serial number ADDR +7 = Flash byte 0

The official document defining EUI-64 is: http://standards.ieee.org/regauth/oui/tutorials/EUI64.html

8.5.3 TLP Processing Hint Requester (TPH) Capability

The PCIe TPH Requester capability is an optional extended capability to support TLP Processing Hints. The following table lists the PCIe TPH extended capability structure for PCIe devices.



Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x1A0	Next Capability Ptr. (0x1C0	Version (0x1)	TPH Capability ID (0x17)	
0x1A4	TPH Requester Capability Register			
0x1A8	TPH Requester Control Register			
0x1AC-0x1B8	TPH ST Table			

8.5.3.1 TPH CAP ID (0x1A0; RO)

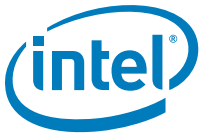
Bit Location	Attribute	Default Value	Description
15:0	RO	0x17	Extended Capability ID PCIe extended capability ID indicating TPH capability.
19:16	RO	0x1	Version Number PCIe TPH extended capability version number.
31:20	RO	0x1C	Next Capability Pointer This field contains the offset to the next PCIe capability structure.

8.5.3.2 TPH Requester Capabilities (0x1A4; RO)

Bit Location	Attribute	Default Value	Description
0	RO	1	No ST Mode Supported: When set indicates the Function is capable of generating Requests without using ST.
1	RO	0	Interrupt Vector Mode Supported: Cleared to indicate that the I210-CS/CL does not support Interrupt Vector Mode of operation.
2	RO	1	Device Specific Mode: Set to indicate that the I210-CS/CL supports Device Specific Mode of operation.
7:3	RO	0	Reserved
8	RO	0	Extended TPH Requester Supported – Cleared to indicate that the function is not capable of generating requests with Extended TPH TLP Prefix.
10:9	RO	01b	ST Table Location – Value indicates if and where the ST Table is located. Defined Encodings are: 00b: ST Table is not present. 01b: ST Table is located in the TPH Requester Capability structure. 10b: ST Table is located in the MSI-X Table structure. 11b: Reserved Default value of 01b indicates that function supports ST table that's located in the TPH Requester Capability structure.
15:11	RO	0x0	Reserved
26:16	RO	0x7	ST_Table Size – System software reads this field to determine the ST_Table_Size N, which is encoded as N-1. The I210-CS/CL supports a table with 8 entries.
31:27	RO	0x0	Reserved

8.5.3.3 TPH Requester Control (0x1A8; R/W)

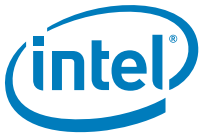
Bit Location	Attribute	Default Value	Description
2:0	RW	0x0	ST Mode Select – Indicates the ST mode of operation selected. The ST mode encodings are as defined below 000b – No Table Mode 001b – Interrupt Vector Mode (not supported by the I210-CS/CL) 010b – Device Specific Mode 0thers – reserved for future use The default value of 000 indicates No Table mode of operation.
7:3	RO	0x0	Reserved
9:8	RW	0x0	TPH Requester Enable: Defined Encodings are: 00b: The I210-CS/CL is not permitted to issue transactions with TPH or Extended TPH as Requester 01b: The I210-CS/CL is permitted to issue transactions with TPH as Requester and is not permitted to issue transactions with Extended TPH as Requester 10b: Reserved 11b: The I210-CS/CL is permitted to issue transactions with TPH and Extended TPH as Requester (the I210-CS/CL does not issue transactions with Extended TPH).
31:10	RO	0x0	Reserved



8.5.3.4 TPH Steering Table (0x1AC - 0x1B8; R/W)

Bit Location	Attribute	Default Value	Description
7:0	RW	0x0	Steering Table Lower Entry 2*n (n = 0...3). A value of zero indicates the tag is not valid
15:8	RO	0x0	Steering Table Upper Entry 2*n (n = 0...3) - RO zero in the I210-CS/CL, as extended tags are not supported.
23:16	RW	0x0	Steering Table Entry 2*n + 1 (n = 0...3) - A value of zero indicates the tag is not valid
31:24	RO	0x0	Steering Table Upper Entry 2*n + 1 (n = 0...3) - RO zero in the I210-CS/CL, as extended tags are not supported.

NOTE: This page intentionally left blank.



9.0 Electrical/Mechanical Specification

9.1 Introduction

These specifications are subject to change without notice.

This chapter describes the I210-CS/CL DC and AC (timing) electrical characteristics. This includes absolute maximum rating, recommended operating conditions, power sequencing requirements, DC and AC timing specifications. The DC and AC characteristics include generic digital 3.3V I/O specification as well as other specifications supported by the I210-CS/CL.

9.2 Operating Conditions

Table 9-1. Absolute Maximum Ratings

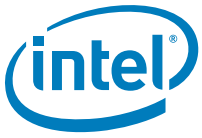
Note: Ratings in these tables are those beyond which permanent device damage is likely to occur. These values should not be used as the limits for normal device operation. Exposure to absolute maximum rating conditions for extended periods might affect device reliability.

Symbol	Parameter	I210-CS/CL (Commercial Temperature SKU)		I210-CS/CL (Industrial Temperature SKU)		Units
		Min	Max	Min	Max	
T_{case}	Case Temperature Under Bias	0	85	-40	105	°C
$T_{storage}$	Storage Temperature Range	-40	125	-40	125	°C
Vi/Vo	3.3V Compatible I/Os Voltage	Vss–0.5	4.0	Vss–0.5	4.0	V
VCC3P3	3.3V DC Supply Voltage	Vss – 0.5	4.0	Vss–0.5	4.0	V

9.2.1 Recommended Operating Conditions

Symbol	Parameter	I210-CS/CL (Commercial Temperature SKU)		I210-CS/CL (Industrial Temperature SKU)		Units	Notes
		Min	Max	Min	Max		
Ta	Operating Temperature Range (Ambient; 0 CFS airflow)	0	70	-40	85	°C	1, 2, 3

1. For normal device operation, adhere to the limits in this table. Sustained operations of a device at conditions exceeding these values, even if they are within the absolute maximum rating limits, may result in permanent device damage or impaired device reliability. Device functionality to stated DC and AC limits is not guaranteed if conditions exceed recommended operating conditions.
2. Recommended operation conditions require accuracy of power supply as defined in Section 9.3.1.
3. With external heat sink. Airflow required for operation in 85 °C ambient temperature.



9.3 Power Delivery

9.3.1 Power Supply Specification

VCC3P3 (3.3V) Parameters				
Parameter	**Description**	**Min**	**Max**	**Units**
Rise Time	Time from 10% to 90% mark	0.1	50	mS
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*V(min)/Rise time (max) Max: 0.8*V(max)/Rise time (min)	24	2880	V/S
Operational Range	Voltage range for normal operating conditions	2.97	3.465	V
Ripple[1]	Maximum voltage ripple (peak to peak)	N/A	70	mV
Overshoot	Maximum overshoot allowed	N/A	100	mV
Overshoot Settling Time	Maximum overshoot allowed duration. (At that time delta voltage should be lower than 5mv from steady state voltage)	N/A	0.05	mS
Decoupling Capacitance	Capacitance range	15		µF
Capacitance ESR	Equivalent series resistance of output capacitance	N/A	50	M Ω
VCC1P5 (1.5V) Parameters				
Parameter	**Description**	**Min**	**Max**	**Units**
Rise Time	Time from 10% to 90% mark	0.1	85	mS
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*V(min)/Rise time (max) Max: 0.8*V(max)/Rise time (min)	14	1440	V/S
Operational Range	Voltage range for normal operating conditions	1.425	1.575	V
Ripple[1]	Maximum voltage ripple (peak to peak)	N/A	40	mV
Overshoot	Maximum overshoot allowed	N/A	100	mV
Overshoot Settling Time	Maximum overshoot allowed duration. (At that time delta voltage should be lower than 5mv from steady state voltage)	N/A	0.1	mS
Decoupling Capacitance	Capacitance range	15		µF
Capacitance ESR	Equivalent series resistance of output capacitance	N/A	50	mΩ
VCC0P9 (0.9V) Parameters				
Parameter	**Description**	**Min**	**Max**	**Units**
Rise Time	Time from 10% to 90% mark	0.1	80	mS
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*V(min)/Rise time (max) Max: 0.8*V(max)/Rise time (min)	7.6	800	V/S
Operational Range	Voltage range for normal operating conditions	0.855	0.945	V
Ripple[1]	Maximum voltage ripple (peak to peak)	N/A	40	mV
Overshoot	Maximum overshoot allowed	N/A	100	mV

Parameter	Description	Min	Max	Units
Overshoot Duration	Maximum overshoot allowed duration. (At that time delta voltage should be lower than 5mv from steady state voltage)	0.0	0.05	mS
Decoupling Capacitance	Capacitance range	15		µF
Capacitance ESR	Equivalent series resistance of output capacitance		50	MΩ

1. Power supply voltage with ripple should not be below minimum power supply operating range.

9.3.1.1 Power On/Off Sequence

On power-on, after 3.3V reaches 90% of its final value, all voltage rails (1.5V and 0.9V) are allowed 20 ms maximum to reach their final operating values. However, to keep leakage current at a minimum, it is recommended to turn on power supplies almost simultaneously (with delay between supplies at most a few milliseconds).

For power-down, it is recommended to turn off all power rails at the same time and let power supply voltage decay.

Table 9-2. Power Sequencing

Symbol	Parameter	Min	Max	Units
T_{3_09}	VCC3P3 (3.3V) power supply stable to VCC0P9 (0.9V) power supply stable	1[3]	20	ms
T_{3_15}	VCC3P3 (3.3V) power supply stable to VCC1P5 (1.5V) power supply stable	1[3]	20	ms
Tm-per	3.3V power supply to PE_RST_N de-assertion[1]	100		ms
Tlpg	Power Supplies Stable to LAN_PWR_GOOD assertion	0		ms
Tlpg-per	LAN_PWR_GOOD assertion to PE_RST_N de-assertion[1]	100		ms
Tper-m	PE_RST_N off before 3.3V power supply down	0		ms
Tlpgw	LAN_PWR_GOOD de-assertion time[2]	1		ms

1. If external LAN_PWR_GOOD is used, this time should be kept between LAN_PWR_GOOD assertion and PE_RST_N de-assertion.
2. Parameter relevant only if external LAN_PWR_GOOD used.
3. With external power supplies.

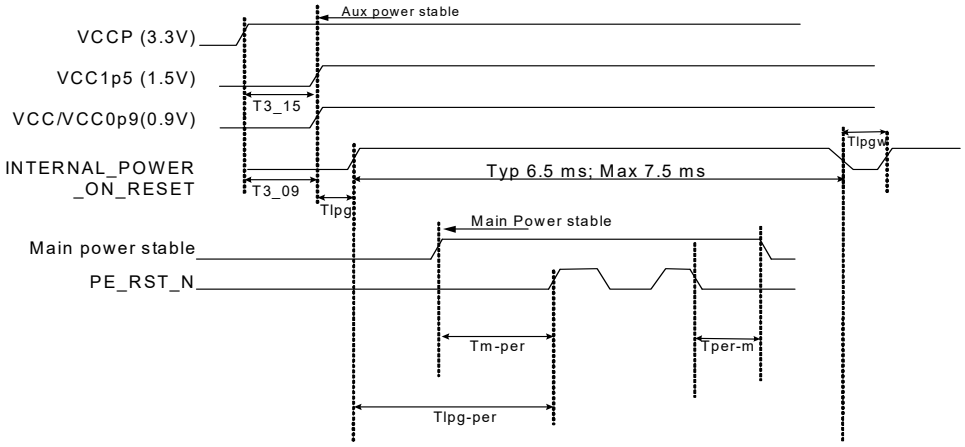


Figure 9-1. Power and Reset Sequencing

9.3.1.2 Power-On Reset Thresholds

The I210-CS/CL internal power-on reset circuitry initiates a full chip reset when voltage levels of power supplies reach certain thresholds at power-up.

Table 9-3. Power-on Reset Thresholds

Symbol	Parameter	Specifications			Units
		Min	Typ	Max	
VTh3.3	Threshold for 3.3 V power supply in power-up	0.96	1.2	1.44	V
VTh0.9	Threshold for 0.9 V power supply in power-up	0.52	0.66	0.8	V
VTh1.5	Threshold for 1.5 V power supply in power-up	0.88	1.1	1.32	V

Note: The POR circuit only generates a reset during the power up but does not monitor the power levels after power is stable. Therefore, it does not generate any reset in power-down or when power levels decrease.

9.4 Ball Summary

See Chapter 2.0 for balls description and ball out map.

9.5 Current Consumption

Condition	Speed (Mb/s)	Condition	Total Power Internal SVR (mW)	0.9V Current-External (mA)	1.5V Current-External (mA)	3.3V Current-External (mA)	Total Power Ext. Regulator (mW)
D0a - active link	1000 fiber	Typ	363	57.9	37.1	61.8	312
	1000 fiber	Max-Commercial	490	111[1]	38[1]	80[1]	421[1]
	1000 fiber	Max-IT	510	121[1]	39[1]	83[1]	441[1]
D0a - idle link	No link	Typ	185	37.1	11.3	33.3	160
	1000 fiber	Typ	354	52	36.8	61.8	306
D3cold - WoL enabled	No link	Typ	104	28.2	11.3	13.4	87
	1000 fiber						
D3cold-WoL disabled (PCIe L3)	No link	Max-Commercial	155				
		Max-IT	173				
D0 uninitialized disabled through DEV_OFF_N	No link	Typ	89	23.3	4.4	13.3	71

Notes:

Typ = Typical units, nominal voltage and room temperature.

Max = Typical units, high voltage, and hot temperature.

9.6 DC/AC Specification

9.6.1 DC Specifications

9.6.1.1 Digital I/O

Table 9-4. Digital IO DC Electrical Characteristics

Symbol	Parameter	Conditions	Min	Max	Units	Note
VCC3P3	Periphery Supply		2.97	3.465	V	3.3V + 5%/3.3V -10%
VCC	Core Supply		0.855	0.945	V	0.9V +/- 5%
VOH	Output High Voltage	IOH = -8 mA; VCC3P3 = Min	2.4		V	3
		IOH = -100 µA; VCC3P3 = Min	VCC3P3-0.2			
VOL	Output Low Voltage	IOL = 8 mA; VCC=Min		0.4	V	4, 5
		IOL = 100 µA; VCC=Min		0.2	V	
VIH	Input High Voltage		0.7 x VCC3P3	VCC3P3 + 0.4	V	1
VIL	Input Low Voltage		-0.4	0.3 x VCC3P3	V	1
Iil	Input Current	VCC3P3 = Max; VI =3.6V/GND		+/- 10	µA	
PU	Internal Pull Up	VIL = 0V	40	150	K Ω	2
	Built-in Hysteresis		150		mV	
Cin	Input Pin Capacitance			5	pF	

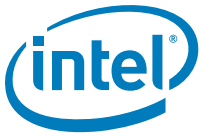


Table 9-4. Digital IO DC Electrical Characteristics (Continued)

Symbol	Parameter	Conditions	Min	Max	Units	Note
Vos	Overshoot		N/A	4	V	
Vus	Undershoot		N/A	-0.4	V	

Notes:

1. Applies to PE_RST_N, LAN_PWR_GOOD, DEV_OFF_N, JTAG_CLK, JTAG_TDI, JTAG_TDO, JTAG_TMS, SDP0,SDP1, SDP2, and SDP3. The input buffer also has hysteresis > 100mV.
2. Internal pull up max characterized at slow corner (125C, VCC3P3=min, process slow); internal pull up min characterized at fast corner (0C, VCC3P3=max, process fast).
3. JTAG_TDO, NVM_CS_N, NVM_SI, NVM_SK, SDP[0], SDP[1], SDP[2], SDP[3], SFP_I2C_DATA - IOH = -6 mA.
4. JTAG_TDO, NVM_CS_N, NVM_SI, NVM_SK, SDP[0], SDP[1], SDP[2], SDP[3], SFP_I2C_DATA - IOL = 6 mA.
5. SFP_I2C_DATA - VOL Max = 0.43 [V].

9.6.1.2 LEDs I/O

Table 9-5. LED IO DC Electrical Characteristics

Symbol	Parameter	Conditions	Min	Max	Units	Note
VCC3P3	Periphery Supply		2.97	3.465	V	
VCC	Core Supply		0.855	0.945	V	
VOH	Output High Voltage	IOH = -20 mA; VCC3P3 = Min	2.4		V	
VOL	Output Low Voltage	IOL = 20 mA; VCC=Min		0.45	V	
VIH	Input High Voltage		0.7 x VCC3P3	VCC3P3 + 0.4	V	1
VIL	Input Low Voltage		-0.4	0.3 x VCC3P3	V	1
Iil	Input Current	VCC3P3 = Max; VI =3.6V/GND		+/- 20	µA	
Vos	Overshoot		N/A	4	V	
Vus	Undershoot		N/A	-0.4	V	

Notes:

1. The input buffer also has hysteresis > 150 mV.
2. Applies to LED0, LED1, and LED2.

9.6.1.3 Open Drain I/Os

Table 9-6. Open Drain DC Specifications (Note 1, 4)

Symbol	Parameter	Condition	Min	Max	Units	Note
VCC3P3	Periphery Supply		2.97	3.465	V	
VCC	Core Supply		0.855	0.945	V	
Vih	Input High Voltage		2.0	5.5	V	
Vil	Input Low Voltage			0.7	V	
Ileakage	Output Leakage Current	0 < Vin < VCC3P3		+/-10	µA	2
Vol	Output Low Voltage	@ Ipullup		0.4	V	4
Iol	Output Low Current	Vol=0.4V	16		mA	

Table 9-6. Open Drain DC Specifications (Note 1, 4) (Continued)

Symbol	Parameter	Condition	Min	Max	Units	Note
Cin	Input Pin Capacitance			5	pF	3
Ioffsmb	Input Leakage Current	VCC3P3 off or floating		+/-10	µA	2

Notes:
1. Applies to SMB_DAT, SMB_CLK, SMB_ALRT_N, PE_WAKE_N and VR_EN pads.
2. Device meets this whether powered or not.
3. Characterized, not tested.
4. OD no high output drive. VOL max=0.4V at 6 mA, VOL max=0.2V at 0.1 mA.

9.6.2 Digital I/F AC Specifications

9.6.2.1 Reset Signals

The timing between the power up sequence and the different reset signals is described in Figure 9-1 and in Table 9-2.

9.6.2.1.1 LAN_PWR_GOOD

The I210-CS/CL uses an internal power on detection circuit in order to generate the LAN_PWR_GOOD signal. Reset can also be implemented when the external power on detection circuit determines that the device is powered up and asserts the LAN_PWR_GOOD signal to reset the device.

9.6.2.2 I^2C AC Specification

Table 9-7 lists the timing of the I2C_CLK and I2C_DATA pins when operating in I^2C mode.

Table 9-7. I^2C Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units
F_{SCL}	I2C_CLK Frequency			100	kHz
T_{BUF}	Time between Stop and Start condition driven by the I210-CS/CL	4.7			µs
$T_{HD:STA}$	Hold Time After Start Condition. After this period, the first clock is generated.	4			µs
$T_{SU:STA}$	Start Condition Setup Time	4.7			µs
$T_{SU:STO}$	Stop Condition Setup Time	4			µs
$T_{HD:DAT}$	Data Hold Time	50[1]			ns
$T_{SU:DAT}$	Data Setup Time	0.25			µs
T_{LOW}	I2C_CLK Low Time	4.7			µs
T_{HIGH}	I2C_CLK High Time	4			µs

1. According to Adesto's AT24C01A/02/04 definition of the 2 wires interface.

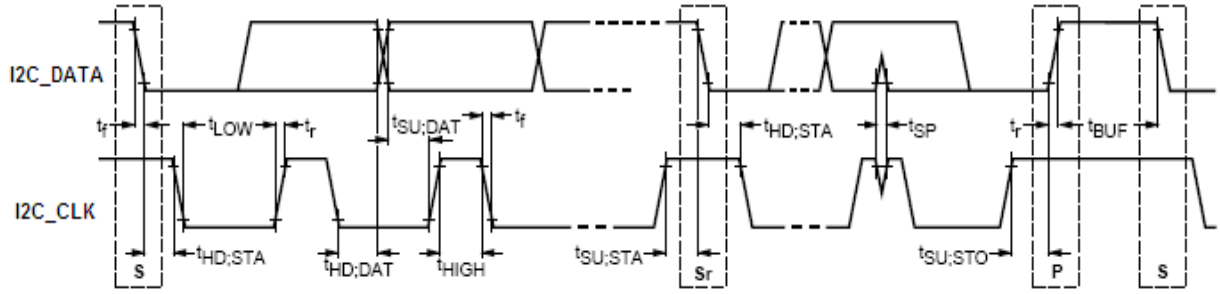


Figure 9-2. I²C I/F Timing Diagram

9.6.2.3 JTAG AC Specification

The I210-CS/CL is designed to support the IEEE 1149.1 standard. Following timing specifications are applicable over recommended operating range from Ta = 0 °C to +70 °C, VCC3P3 = 3.3V, Cload = 16 pF (unless otherwise noted). For JTAG I/F timing specification see Table 9-8 and Figure 9-6.

Table 9-8. JTAG I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{JCLK}	JTCK clock frequency			10	MHz	
t_{JH}	JTMS and JTDI hold time	10			nS	
t_{JSU}	JTMS and JTDI setup time	10			nS	
t_{JPR}	JTDO propagation Delay			15	nS	

Notes:
1. Table 9-8 applies to JTCK, JTMS, JTDI and JTDO.
2. Timing measured relative to JTCK reference voltage of VCC3P3/2.

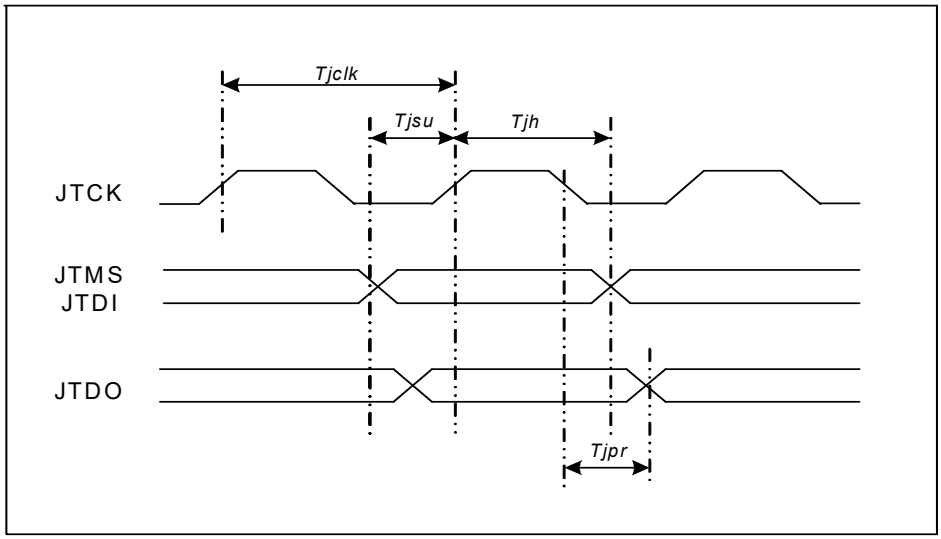


Figure 9-3. JTAG AC Timing Diagram

9.6.2.4 MDIO AC Specification

The I210-CS/CL is designed to support the MDIO specifications defined in IEEE 802.3 clause 22. Following timing specifications are applicable over recommended operating range from Ta = 0 ^{o}C to +70 ^{o}C
(-40 ^{o}C to +85 ^{o}C for industrial temperature SKUs), VCC3P3 = 3.3V, Cload = 16 pF (unless otherwise noted). For MDIO I/F timing specification see Table 9-9.

Table 9-9. MDIO I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{MCLK}	MDC clock frequency			2	MHz	
t_{MH}	MDIO hold time	10			nS	
t_{MSU}	MDIO setup time	10			nS	
t_{MPR}	MDIO propagation Delay	10		300	nS	

Notes:
1. Table 9-9 applies to MDIO0, MDC0, MDIO1, MDC1, MDIO2, MDC2, MDIO3, and MDC3.
2. Timing measured relative to MDC reference voltage of 2.0V (Vih).

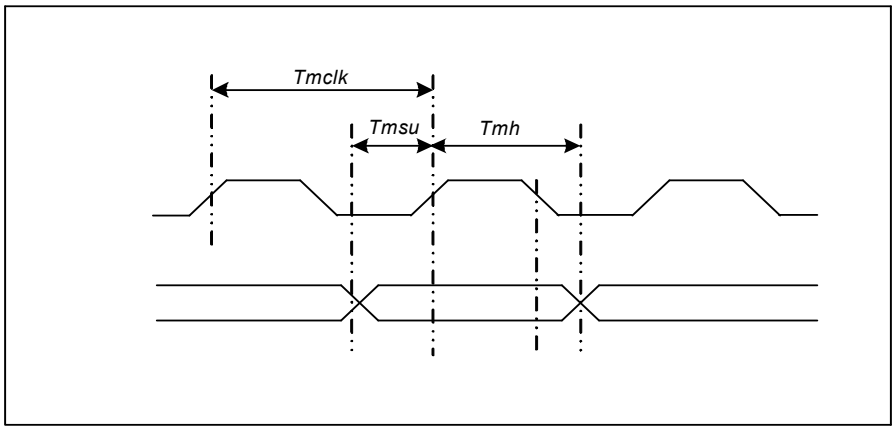


Figure 9-4. MDIO Input AC Timing Diagram

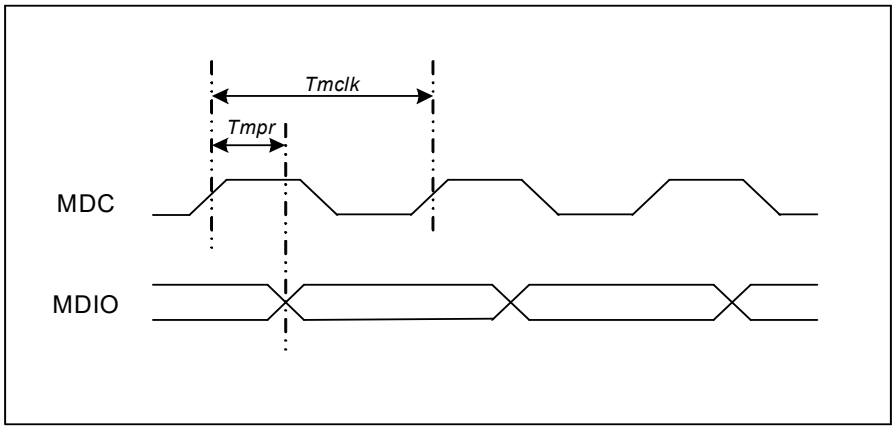


Figure 9-5. MDIO Output AC Timing Diagram

9.6.2.5 SFP 2 Wires I/F AC Specification

According to Adesto's AT24C01A/02/04 definition of the 2 wires I/F bus.

9.6.2.6 PCIe Interface DC/AC Specification

The I210-CS/CL PCIe Gen 1 interface supports the electrical specifications defined in:

- PCI Express* 2.0 Card Electro-Mechanical (CEM) Specification.
- PCI Express* 2.1 Base Specification, Chapter 4.

9.6.2.6.1 PCIe Specification - Input Clock

The input clock for PCIe must be a differential input clock in frequency of 100 MHz. For full specifications please check the PCI Express* 2.0 Card Electro-Mechanical (CEM) Specification (refclk specifications for Gen 1).

9.6.3 SerDes DC/AC Specification

The SerDes interface supports the following standards:

1. PICMG 3.1 specification Rev 1.0 1000BASE-BX.
2. 1000BASE-KX electrical specification defined IEEE802.3ap clause 70.
3. SGMII on 1000BASE-BX or 1000BASE-KX compliant electrical interface (AC coupling with internal clock recovery).
4. SFP (Small Form factor Pluggable) Transceiver Rev 1.0

9.6.4 PHY Specification

The specifications define the interface for the back-plane board connection, interface to external 1000BASE-T PHY and the interface to fiber or SFP module.

DC/AC specification is according to Standard 802.3 and 802.3ab version 2008.

9.6.5 XTAL/Clock Specification

The 25 MHz reference clock of the I210-CS/CL can be supplied either from a crystal or from an external oscillator. The recommended solution is to use a crystal.

9.6.5.1 Crystal Specification

Table 9-10. Specification for External Crystal

Parameter Name	Symbol	Recommended Value	Conditions
Frequency	f_o	25.000 [MHz]	@25 [°C]
Vibration mode		Fundamental	
Cut		AT	
Operating /Calibration Mode		Parallel	
Frequency Tolerance @25°C	$\Delta f/f_o$ @25°C	±30 [ppm]	@25 [°C]
Temperature Tolerance	$\Delta f/f_o$	±30 [ppm]	
Operating Temperature	T_{opr}	0 to +70 [°C] -40 to +85 [°C]	Commercial grade Industrial grade
Non Operating Temperature Range	T_{opr}	-30 to +85 [°C]	
Equivalent Series Resistance (ESR)	R_s	50 [Ω] maximum	@25 [MHz]
Shunt Capacitance	C_o	6 [pF] maximum	
Load Capacitance	C_{load}	16 to 18 pF	
Max Drive Level	D_L	0.5 [mW]	
Aging	$\Delta f/f_o$	±5 [ppm/year]	
External Capacitors	C_1, C_2	27 [pF]	

9.6.5.2 External Clock Oscillator Specifications

When using an external oscillator the following connection must be used.

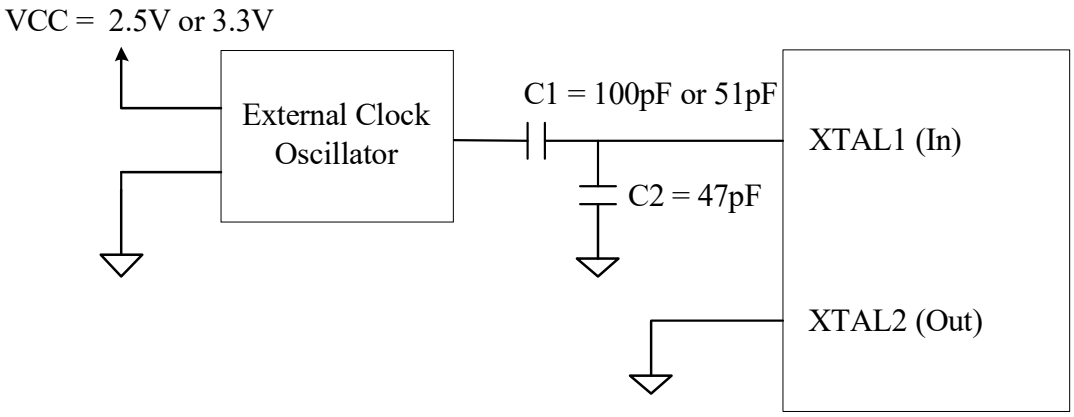


Figure 9-6. External Clock Oscillator Connectivity to The I210-CS/CL

Table 9-11. Specification for XTAL1 (In)

Parameter Name	Symbol	Value	Conditions
Voltage Input High (minimum)	VIH (min)	1.4 [V]	
Voltage Input High (maximum)	VIH (max)	2.0 [V]	
Target XTAL1 (In) amplitude	VIH (typ)	1.7 [V]	
Voltage Input Low (maximum)	VIL (max)	200 [mV]	
Input Impedance		High impedance	

Table 9-12. Specification for External Clock Oscillator

Parameter Name	Symbol	Value	Conditions
Frequency	f_o	25.0 [MHz]	@25 [°C]
External OSC Supply Swing	V_{p-p}	2.5 ± 0.25 [V] or 3.3 ± 0.33 [V]	
Frequency Tolerance	$\Delta f/f_o$	±50 [ppm]	-20 to +70 [°C]
Operating Temperature	T_{opr}	0 to +70 [°C] -40 to +85 [°C]	Commercial grade Industrial grade
Maximum jitter 12KHz-20 MHz RMS[1]		1.5 [ps]	

1. At the XTAL1 input.

9.6.6 Switching Voltage Regulator (SVR) Capacitor Electrical Specifications

The following table lists the electrical performance of the 0.9V/1.5V SVR.

Parameter	Min	Typ	Max	Unit	Comments
Regulator input voltage	2.97	3.3	3.465	V	
Regulator output voltages		0.9 1.5		V	
Output Voltage Accuracy		5		%	Not including line and load regulation errors.
Load Current	0		175	mA	175mA (max) for each 0.9V and 1.5V rail.
Startup Time	4	5	6	ms	
Load Capacitor		20		µF	Ceramic bulk capacitors.
Flying Capacitor		39		nF	Located close to package related pins.

9.7 Package

The I210-CS/CL is assembled in one, single package type: 9 mm x 9 mm 64-pin QFN package.

9.7.1 Mechanical Specification for the 9 x 9 QFN Package

Table 9-13. I210-CS/CL 9 x 9 Package Mechanical Specifications

Body Size	Pin Count	Pin Pitch	Ball Matrix	Center Matrix	Substrate
9 x 9	64	0.5 mm	N/A, peripheral	N/A, exposed pad	N/A, led frame-based package

9.7.2 9 x 9 QFN Package Schematics

Refer to Figure 2-2.

9.8 Flash Devices

While Intel does not make recommendations regarding these devices, the following devices might match to be used successfully. Minimum Flash size required is 8 Mb or 16 Mb, depending if an Expansion/Option ROM module is needed.

9.8.1 Flash

Type: SPI Flash

Supported Flash Parts
Winbond* Compatible:[1] W25X40BVSNIG W25X80BVSNIG W25Q40BVSNIG W25Q80BVSNIG W25Q16 W25Q32 **Winbond Validated:[2]** W25Q80BVSSIG. W25Q16CVSSIG. W25X16A.
Micron* (Numonyx*) Compatible:[1] M25PE40 M25PE80 M25PX80 N25Q032 **Micron (Numonyx) Validated:[2]** M25PX80VMW6G. M25PE80VMW6TG. N25Q032A13ESE40F.
Macronix* Compatible:[1] MX25L4005 MX25L8005 MX25L1633E. MX25L1633EM2I. **Macronix Validated:[2]** MX25L1633EM2I.
Microchip (SST)* Compatible:[1] SST25VF032B SST25VF040B-80-4I-SAE SST25VF080B-80-4I-SAE **Microchip (SST) Validated:[2]** SST25VF040B.
Adesto Compatible:[1] AT25DF041A **Adesto Validated:[2]** AT25DF081A). AT25DF321A
EON* Validated:[2] EN25Q32B EN25QH16 EN25Q64
Fidelix* Validated:[2] FM25S16A
AMIC* Validated:[2] A25L040M

Supported Flash Parts
Spansion* Validated:[2] S25FL008K0XMF
ESMT* Validated:[2] F25L04PA

1. Compatible by design but not tested.
2. Validated Flash parts.

10.0 Design Considerations

This section provides general design considerations and recommendations when selecting components and connecting special pins to the I210-CS/CL. Intel recommends that these design considerations be used in conjunction with the following board design documents:

- Intel® Ethernet Controller I210-IS – Schematics / Diagrams
- Intel® Ethernet Controller I210-IS – Layout Review Checklist
- Intel® Ethernet Controller I210-AT/IT – Layout Review Checklist

10.1 PCIe

10.1.1 Port Connection to the I210-CS/CL

PCIe is a dual simplex point-to-point serial differential low-voltage interconnect with a signaling bit rate of 2.5 Gb/s per direction. The I210-CS/CL's PCIe port consists of an integral group of transmitters and receivers. The link between the PCIe ports of two devices is a x1 lane that also consists of a transmitter and a receiver pair. Note that each signal is 8b/10b encoded with an embedded clock.

The PCIe topology consists of a transmitter (Tx) located on one device connected through a differential pair connected to the receiver (Rx) on a second device. The I210-CS/CL can be located on a LOM or on an add-in card using a connector specified by PCIe.

The lane is AC-coupled between its corresponding transmitter and receiver. The AC-coupling capacitor is located on the board close to transmitter side. Each end of the link is terminated on the die into nominal 100 Ω differential DC impedance. Board termination is not required.

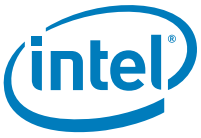
For more information on PCIe, refer to the *PCI Express* Base Specification, Revision 1.1*, *PCI Express* Card Electromechanical Specification, Revision 1.1RD, and PCIe v2.1 (2.5GT/s) Gen1 x 1.*

For information about the I210-CS/CL's PCIe power management capabilities, see Section 5.0.

10.1.2 PCIe Reference Clock

The I210-CS/CL uses a 100 MHz differential reference clock, denoted PECLKp and PECLKn. This signal is typically generated on the system board and routed to the PCIe port. For add-in cards, the clock is furnished at the PCIe connector.

The frequency tolerance for the PCIe reference clock is +/- 300 ppm.



10.1.3 Other PCIe Signals

The I210-CS/CL also implements other signals required by the PCIe specification. The I210-CS/CL signals power management events to the system using the PE_WAKE_N signal, which operates very similarly to the familiar PCI PME# signal. Finally, there is a PE_RST_N signal, which serves as the familiar reset function for the I210-CS/CL.

10.1.4 PCIe Routing

Contact your Intel representative for information regarding the PCIe signal routing.

10.2 Clock Source

All designs require a 25 MHz clock source. The I210-CS/CL uses the 25 MHz source to generate clocks up to 125 MHz and 1.25 GHz for the PHY circuits. For optimum results with lowest cost, connect a 25 MHz parallel resonant crystal and appropriate load capacitors at the XTAL1 and XTAL2 leads. The frequency tolerance of the timing device should be 30 ppm or better. Refer to the Intel® Ethernet Controllers Timing Device Selection Guide for more information on choosing crystals.

For further information regarding the clock for the I210-CS/CL, refer to the sections about frequency control, crystals, and oscillators that follow.

10.2.1 Frequency Control Device Design Considerations

This section provides information regarding frequency control devices, including crystals and oscillators, for use with all Intel Ethernet controllers. Several suitable frequency control devices are available; none of which present any unusual challenges in selection. The concepts documented herein are applicable to other data communication circuits, including Platform LAN Connect devices (PHYs).

Intel Ethernet controllers contain amplifiers, which when used with the specific external components, form the basis for feedback oscillators. These oscillator circuits, which are both economical and reliable, are described in more detail in Section 10.3.1.

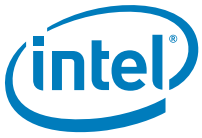
The chosen frequency control device vendor should be consulted early in the design cycle. Crystal and oscillator manufacturers familiar with networking equipment clock requirements can provide assistance in selecting an optimum, low-cost solution.

10.2.2 Frequency Control Component Types

Several types of third-party frequency reference components are currently marketed. A discussion of each follows, listed in preferred order.

10.2.2.1 Quartz Crystal

Quartz crystals are generally considered to be the mainstay of frequency control components due to their low cost and ease of implementation. They are available from numerous vendors in many package types and with various specification options.



10.2.2.2 Fixed Crystal Oscillator

A packaged fixed crystal oscillator comprises an inverter, a quartz crystal, and passive components conveniently packaged together. The device renders a strong, consistent square wave output. Oscillators used with microprocessors are supplied in many configurations and tolerances.

Crystal oscillators should be restricted to use in special situations, such as shared clocking among devices or multiple controllers. As clock routing can be difficult to accomplish, it is preferable to provide a separate crystal for each device.

10.2.2.3 Programmable Crystal Oscillators

A programmable oscillator can be configured to operate at many frequencies. The device contains a crystal frequency reference and a phase lock loop (PLL) clock generator. The frequency multipliers and divisors are controlled by programmable fuses.

A programmable oscillator's accuracy depends heavily on the Ethernet device's differential transmit lines. The Physical Layer (PHY) uses the clock input from the device to drive a differential Manchester (for 10 Mb/s operation), an MLT-3 (for 100 Mbps operation) or a PAM-5 (for 1000 Mb/s operation) encoded analog signal across the twisted pair cable. These signals are referred to as self-clocking, which means the clock must be recovered at the receiving link partner. Clock recovery is performed with another PLL that locks onto the signal at the other end.

PLLs are prone to exhibit frequency jitter. The transmitted signal can also have considerable jitter even with the programmable oscillator working within its specified frequency tolerance. PLLs must be designed carefully to lock onto signals over a reasonable frequency range. If the transmitted signal has high jitter and the receiver's PLL loses its lock, then bit errors or link loss can occur.

PHY devices are deployed for many different communication applications. Some PHYs contain PLLs with marginal lock range and cannot tolerate the jitter inherent in data transmission clocked with a programmable oscillator. The American National Standards Institute (ANSI) X3.263-1995 standard test method for transmit jitter is not stringent enough to predict PLL-to-PLL lock failures, therefore, the use of programmable oscillators is not recommended.

10.2.2.4 Ceramic Resonator

Similar to a quartz crystal, a ceramic resonator is a piezoelectric device. A ceramic resonator typically carries a frequency tolerance of ±0.5%, – inadequate for use with Intel Ethernet controllers, and therefore, should not be utilized.

10.3 Crystal Support

10.3.1 Crystal Selection Parameters

All crystals used with Intel Ethernet controllers are described as AT-cut, which refers to the angle at which the unit is sliced with respect to the long axis of the quartz stone. Table 10-14 lists crystals which have been used successfully in other designs (however, no particular product is recommended):

Table 10-14. Crystal Manufacturers and Part Numbers

Manufacturer	Part No.
KDS America	DSX321G
NDK America Inc.	41CD25.0F1303018
TXC Corporation - USA	7A25000165 9C25000008

For information about crystal selection parameters, see Section 9.6.5 and Table 9-10.

10.3.1.1 Vibrational Mode

Crystals in the above-referenced frequency range are available in both fundamental and third overtone. Unless there is a special need for third overtone, use fundamental mode crystals.

At any given operating frequency, third overtone crystals are thicker and more rugged than fundamental mode crystals. Third overtone crystals are more suitable for use in military or harsh industrial environments. Third overtone crystals require a trap circuit (extra capacitor and inductor) in the load circuitry to suppress fundamental mode oscillation as the circuit powers up. Selecting values for these components is beyond the scope of this document.

10.3.1.2 Nominal Frequency

Intel Ethernet controllers use a crystal frequency of 25.000 MHz. The 25 MHz input is used to generate a 125 MHz transmit clock for 100BASE-TX and 1000BASE-TX operation – 10 MHz and 20 MHz transmit clocks, for 10BASE-T operation.

10.3.1.3 Frequency Tolerance

The frequency tolerance for an Ethernet Platform LAN Connect is dictated by the IEEE 802.3 specification as ±50 parts per million (ppm). This measurement is referenced to a standard temperature of 25° C. Intel recommends a frequency tolerance of ±30 ppm.

10.3.1.4 Temperature Stability and Environmental Requirements

Temperature stability is a standard measure of how the oscillation frequency varies over the full operational temperature range (and beyond). Several optional temperature ranges are currently available, including -40° C to +85° C for industrial environments. Some vendors separate operating temperatures from temperature stability. Manufacturers may also list temperature stability as 50 ppm in their data sheets.

Note: Crystals also carry other specifications for storage temperature, shock resistance, and reflow solder conditions. Crystal vendors should be consulted early in the design cycle to discuss the application and its environmental requirements.

10.3.1.5 Crystal Oscillation Mode

The terms series-resonant and parallel-resonant are often used to describe crystal oscillator circuits. Specifying parallel mode is critical to determining how the crystal frequency is calibrated at the factory.

A crystal specified and tested as series resonant oscillates without problem in a parallel-resonant circuit, but the frequency is higher than nominal by several hundred parts per million. The purpose of adding load capacitors to a crystal oscillator circuit is to establish resonance at a frequency higher than the crystal's inherent series resonant frequency.

Figure 10-7 shows the recommended placement and layout of an internal oscillator circuit. Note that pin X1 and X2 refers to XTAL1 and XTAL2 in the Ethernet device, respectively. The crystal and the capacitors form a feedback element for the internal inverting amplifier. This combination is called parallel-resonant, because it has positive reactance at the selected frequency. In other words, the crystal behaves like an inductor in a parallel LC circuit. Oscillators with piezoelectric feedback elements are also known as "Pierce" oscillators.

10.3.1.6 Load Capacitance and Discrete Capacitors

The formula for crystal load capacitance is as follows:

$$C_L = \frac{(C1 \cdot C2)}{(C1 + C2)} + C_{stray}$$

where:

C_L is the rated C_{load} of the crystal component and C1 and C2 are discrete crystal circuit capacitors.
C_{stray} allows for additional capacitance from solder pads, traces and the I210-CS/CL package. Individual stray capacitance components can be estimated and added as parallel capacitances. Note that total C_{stray} is typically 3 pF to 7 pF.

Solve for the discrete capacitor values as follows:
$C1 = C2 = 2 * [C_{load} - C_{stray}]$

For example:

If total C_{stray} = 4.0 pF and if the C_{load} rating is 18 pF, then the calculated C1 and C2 = 2 * [18 pF - 4.0 pF] = 28 pF.

Note: Because 28 pF is not a standard value, use 27 pF capacitors for C1 and C2, which is the closest standard value.

The oscillator frequency should be measured with a precision frequency counter where possible. The values of C1 and C2 should be fine tuned for the design. As the actual capacitive load increases, the oscillator frequency decreases.

Note: Intel recommends COG or NPO capacitors with a tolerance of ±5% (approximately ±1 pF) or smaller.

10.3.1.7 Shunt Capacitance

The shunt capacitance parameter is relatively unimportant compared to load capacitance. Shunt capacitance represents the effect of the crystal's mechanical holder and contacts. The shunt capacitance should equal a maximum of 6 pF.

10.3.1.8 Equivalent Series Resistance

Equivalent Series Resistance (ESR) is the real component of the crystal's impedance at the calibration frequency, which the inverting amplifier's loop gain must overcome. ESR varies inversely with frequency for a given crystal family. The lower the ESR, the faster the crystal starts up. Use crystals with an ESR value of 50 Ω or better.

10.3.1.9 Drive Level

Drive level refers to power dissipation in use. The allowable drive level for a Surface Mounted Technology (SMT) crystal is less than its through-hole counterpart, because surface mount crystals are typically made from narrow, rectangular AT strips, rather than circular AT quartz blanks.

Some crystal data sheets list crystals with a maximum drive level of 1 mW. However, Intel Ethernet controllers drive crystals to a level less than the suggested 0.3 mW value. This parameter does not have much value for on-chip oscillator use.

10.3.1.10 Aging

Aging is a permanent change in frequency (and resistance) occurring over time. This parameter is most important in its first year because new crystals age faster than old crystals. Use crystals with a maximum of ±5 ppm per year aging.

10.3.1.11 Reference Crystal

The normal tolerances of the discrete crystal components can contribute to small frequency offsets with respect to the target center frequency. To minimize the risk of tolerance-caused frequency offsets causing a small percentage of production line units to be outside of the acceptable frequency range, it is important to account for those shifts while empirically determining the proper values for the discrete loading capacitors, C1 and C2.

Even with a perfect support circuit, most crystals will oscillate slightly higher or slightly lower than the exact center of the target frequency. Therefore, frequency measurements (which determine the correct value for C1 and C2) should be performed with an ideal reference crystal. When the capacitive load is exactly equal to the crystal's load rating, an ideal reference crystal will be perfectly centered at the desired target frequency.

10.3.1.11.1 Reference Crystal Selection

There are several methods available for choosing the appropriate reference crystal:

- If a Saunders and Associates (S&A) crystal network analyzer is available, then discrete crystal components can be tested until one is found with zero or nearly zero ppm deviation (with the appropriate capacitive load). A crystal with zero or near zero ppm deviation will be a good reference crystal to use in subsequent frequency tests to determine the best values for C1 and C2.
- If a crystal analyzer is not available, then the selection of a reference crystal can be done by measuring a statistically valid sample population of crystals, which has units from multiple lots and approved vendors. The crystal, which has an oscillation frequency closest to the center of the distribution, should be the reference crystal used during testing to determine the best values for C1 and C2.
- It may also be possible to ask the approved crystal vendors or manufacturers to provide a reference crystal with zero or nearly zero deviation from the specified frequency when it has the specified cload capacitance.

When choosing a crystal, customers must keep in mind that to comply with IEEE specifications for 10/100 and 10/100/1000Base-T Ethernet LAN, the transmitter reference frequency must be precise within ±50 ppm. Intel recommends customers to use a transmitter reference frequency that is accurate to within ±30 ppm to account for variations in crystal accuracy due to crystal manufacturing tolerance.

10.3.1.11.2 Circuit Board

Since the dielectric layers of the circuit board are allowed some reasonable variation in thickness, the stray capacitance from the printed board (to the crystal circuit) will also vary. If the thickness tolerance for the outer layers of dielectric are controlled within ±17 percent of nominal, then the circuit board should not cause more than ±2 pF variation to the stray capacitance at the crystal. When tuning crystal frequency, it is recommended that at least three circuit boards are tested for frequency. These boards should be from different production lots of bare circuit boards.

Alternatively, a larger sample population of circuit boards can be used. A larger population will increase the probability of obtaining the full range of possible variations in dielectric thickness and the full range of variation in stray capacitance.

Next, the exact same crystal and discrete load capacitors (C1 and C2) must be soldered onto each board, and the LAN reference frequency should be measured on each circuit board.

The circuit board, which has a LAN reference frequency closest to the center of the frequency distribution, should be used while performing the frequency measurements to select the appropriate value for C1 and C2.

10.3.1.11.3 Temperature Changes

Temperature changes can cause the crystal frequency to shift. Therefore, frequency measurements should be done in the final system chassis across the system's rated operating temperature range.

10.3.2 Crystal Placement and Layout Recommendations

Crystal clock sources should not be placed near I/O ports or board edges. Radiation from these devices can be coupled into the I/O ports and radiate beyond the system chassis. Crystals should also be kept away from the Ethernet magnetics module to prevent interference.

Note: Failure to follow these guidelines could result in the 25 MHz clock failing to start.

When designing the layout for the crystal circuit, the following rules must be used:

- Place load capacitors as close as possible (within design-for-manufacturability rules) to the crystal solder pads. They should be no more than 90 mils away from crystal pads.
- The two load capacitors, crystal component, the Ethernet controller device, and the crystal circuit traces must all be located on the same side of the circuit board (maximum of one via-to-ground load capacitor on each XTAL trace).
- Use 27 pF (5% tolerance) 0402 load capacitors.
- Place load capacitor solder pad directly in line with circuit trace (see Figure 10-7, point A).
- Use 50 Ω impedance single-ended microstrip traces for the crystal circuit.
- Route traces so that electro-magnetic fields from XTAL2 do not couple onto XTAL1. Do not route as differential traces.
- Route XTAL1 and XTAL2 traces to nearest inside corners of crystal pad (see Figure 10-7, point B).
- Ensure that the traces from XTAL1 and XTAL2 are symmetrically routed and that their lengths are matched.
- The total trace length of XTAL1 or XTAL2 should be less than 750 mils.

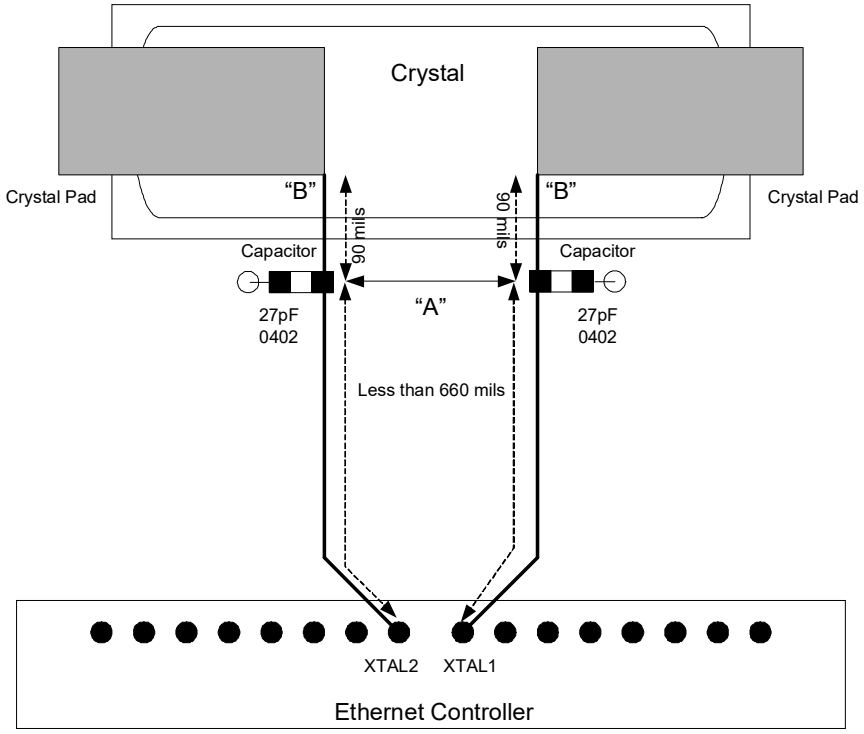
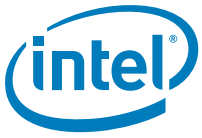


Figure 10-7. Recommended Crystal Placement and Layout

10.4 Oscillator Support

The I210-CS/CL clock input circuit is optimized for use with an external crystal. However, an oscillator can also be used in place of the crystal with the proper design considerations (see Table 9-12 for detail clock oscillator specifications):



- The input capacitance introduced by the I210-CS/CL (approximately 20 pF) is greater than the capacitance specified by a typical oscillator (approximately 15 pF).
- The input clock jitter from the oscillator can impact the I210-CS/CL clock and its performance.

Note: The power consumption of additional circuitry equals about 1.5 mW.

Table 10-15 lists oscillators that can be used with the I210-CS/CL. Please note that no particular oscillator is recommended):

Table 10-15. Oscillator Manufacturers and Part Numbers

Manufacturer	Part No.
NDK AMERICA INC	2560TKA-25M
TXC CORPORATION - USA	6N25000160 or 7W25000025
CITIZEN AMERICA CORP	CSX750FJB25.000M-UT
Raltron Electronics Corp	CO4305-25.000-T-TR
MtronPTI	M214TCN
Kyocera Corporation	KC5032C-C3

10.4.1 Oscillator Placement and Layout Recommendations

Oscillator clock sources should not be placed near I/O ports or board edges. Radiation from these devices can be coupled into the I/O ports and radiate beyond the system chassis. Oscillators should also be kept away from the Ethernet magnetics module to prevent interference.

10.5 I210-CS/CL Power Supplies

The I210-CS/CL requires three power rails: 3.3 Vdc, 1.5 Vdc, and 0.9 Vdc. Intel recommends that board designers use the integrated switching voltage regulators derived from a single 3.3 Vdc supply to reduce Bill of Material (BOM) costs. A central power supply can provide the required voltage sources designed by a system power engineer. If the LAN wake capability is used, all voltages must remain present during system power down. External voltage regulators need to generate the proper voltage, supply current requirements (with adequate margin), and provide the proper power sequencing.

Refer to Section 10.5.2 for detailed information about power supply sequencing rules.

10.5.1 Power Delivery Solutions

Figure 10-8 shows the intended design options for power delivery solutions. See Section 9.3 for more details.

Note: Follow the power sequencing instructions described in Section 10.5.2.

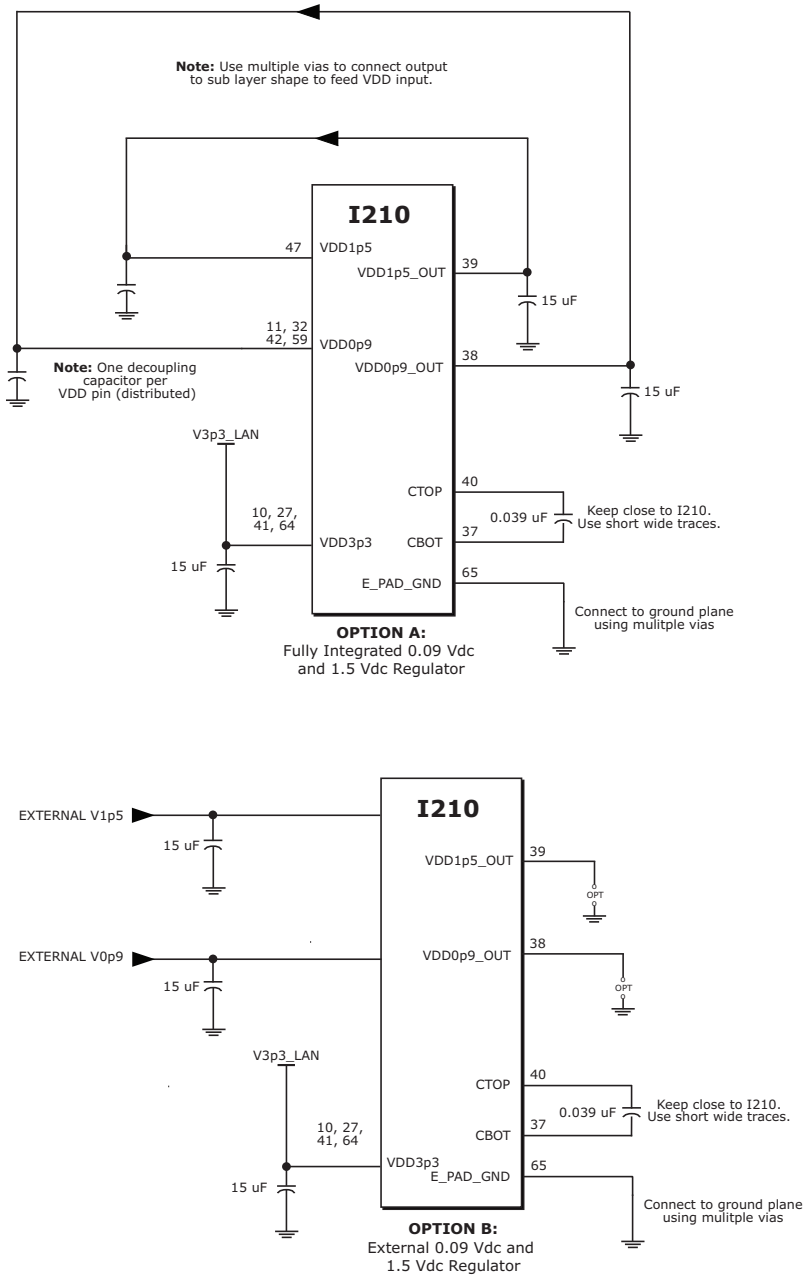


Figure 10-8. Power Delivery Solutions

10.5.2 Ethernet Controller I210-CS/CL Power Sequencing

Designs must comply with power sequencing requirements to avoid latch-up and forward-biased internal diodes (see Figure 10-9).

The general guideline for sequencing is:

1. Power up the 3.3 Vdc rail.
2. Power up the 1.5 Vdc next.
3. Power up the 0.9 Vdc rail last.

For power down, there is no requirement (only charge that remains is stored in the decoupling capacitors).

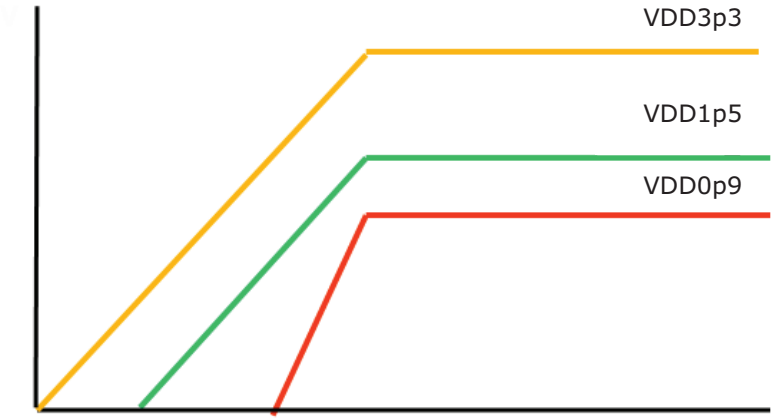


Figure 10-9. Power Sequencing Guideline

10.5.2.1 Power Up Sequence (External Voltage Regulator)

The board designer controls the power up sequence with the following stipulations (see Figure 10-10):

- 1.5 Vdc must not exceed 3.3 Vdc by more than 0.3 Vdc.
- 0.9 Vdc must not exceed 1.5 Vdc by more than 0.3 Vdc.
- 0.9 Vdc must not exceed 3.3 Vdc by more than 0.3 Vdc.

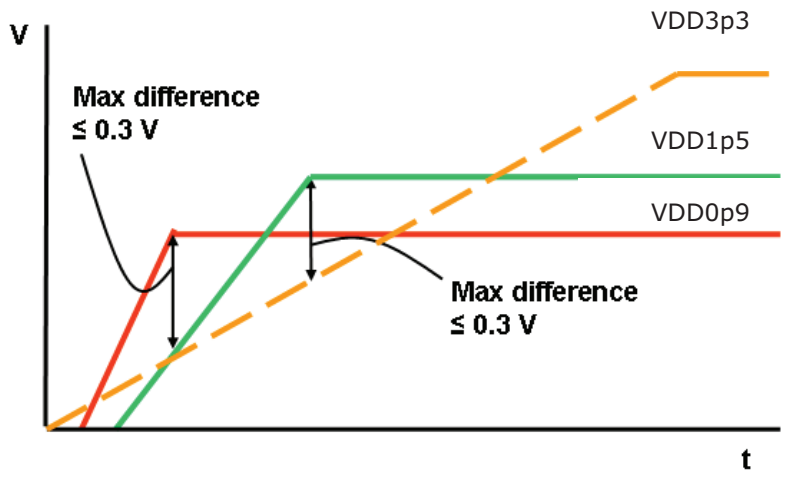


Figure 10-10.External Voltage Regulator Power-up Sequence

10.5.2.2 Power Up-Sequence (Internal SVR)

The I210-CS/CL controls the power-up sequence internally and automatically with the following conditions (see Figure 10-11):

- 3.3 Vdc must be the source for the internal LVR.
- 1.5 Vdc never exceeds 3.3 Vdc.
- 0.9 Vdc never exceeds 3.3 Vdc or 1.5 Vdc.

The ramp is delayed internally, with T_{delay} depending on the rising slope of the 3.3 Vdc ramp (see Table 11-2).

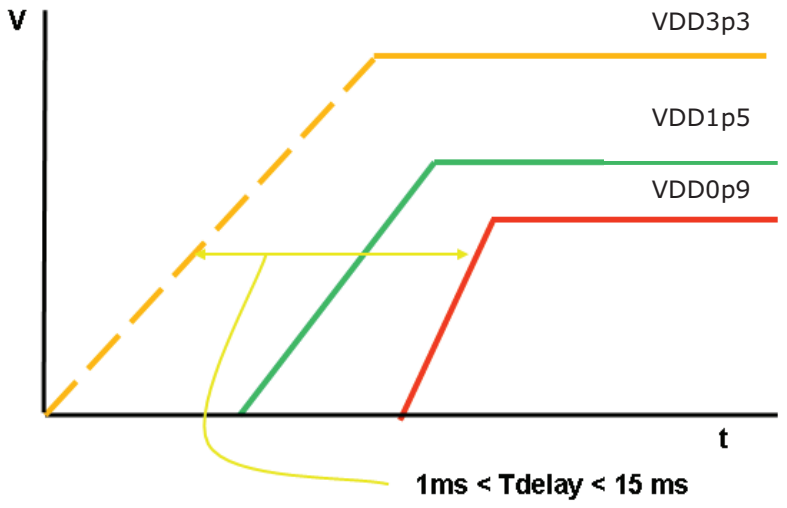


Figure 10-11.Internal SVR Power-Up Sequence

10.5.3 Power and Ground Planes

Good grounding requires minimizing inductance levels in the interconnections and keeping ground returns short, signal loop areas small, and power inputs bypassed to signal return, will significantly reduce EMI radiation.

The following guidelines help reduce circuit inductance in both backplanes and motherboards:

- Route traces over a continuous plane with no interruptions. Do not route over a split power or ground plane. If there are vacant areas on a ground or power plane, avoid routing signals over the vacant area. This increases inductance and EMI radiation levels.
- Separate noisy digital grounds from analog grounds to reduce coupling. Noisy digital grounds may affect sensitive DC subsystems.
- All ground vias should be connected to every ground plane; and every power via should be connected to all power planes at equal potential. This helps reduce circuit inductance.
- Physically locate grounds between a signal path and its return. This minimizes the loop area.
- Avoid fast rise/fall times as much as possible. Signals with fast rise and fall times contain many high frequency harmonics, which can radiate EMI.
- The ground plane beneath a magnetics module should be split. The RJ45 connector side of the transformer module should have chassis ground beneath it.
- Power delivery traces should be a minimum of 20 mils wide at all places from the source to the destination with neck down at package pins. The distribution of power is better done with a copper plane or shape under the PHY. This provides low inductance connectivity to decoupling capacitors. Decoupling capacitors should be placed as close as possible to the point of use and should avoid sharing vias with other decoupling capacitors. Decoupling capacitor placement control should be done for the PHY as well as any external regulators if used.
- An SVR fly capacitor should be preferentially placed near pin 37 and 40 with wide traces to limit in-line inductance.
- SVR output routing: AVDD09_VR_O (pin 38) should be connected with wide traces and plane shape using more than one via for a layer change to VDD09 pins. The net should have recommended bulk and decoupling capacitance strongly joined into this route. AVDD15_VR_O (pin 39) has similar requirements but has lower currents so it might require only wide traces and a single via for any layer change.

10.6 Device Disable

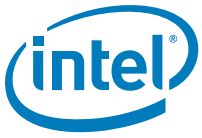
For a LOM design, it might be desirable for the system to provide BIOS-setup capability for selectively enabling or disabling LOM devices. This enables designers more control over system resource-management, avoid conflicts with add-in NIC solutions, etc. The I210-CS/CL provides support for selectively enabling or disabling it.

Device disable is initiated by asserting the asynchronous DEV_OFF_N pin. The DEV_OFF_N pin has an internal pull-up resistor, so that it can be left not connected to enable device operation.

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

Assertion or deassertion of PCIe PE_RST_N does not have any effect while the I210-CS/CL is in device disable mode (that is, the I210-CS/CL stays in the respective mode as long as DEV_OFF_N is asserted). However, the I210-CS/CL might momentarily exit the device disable mode from the time PCIe PE_RST_N is de-asserted again and until the Flash is read.

During power-up, the DEV_OFF_N pin is ignored until the NVM is read. From that point, the I210-CS/CL might enter device disable if DEV_OFF_N is asserted.



Note: The DEV_OFF_N pin should maintain its state during system reset and system sleep states. It should also insure the proper default value on system power up. For example, a designer could use a GPIO pin that defaults to 1b (enable) and is on system suspend power. For example, it maintains the state in S0-S5 ACPI states).

10.6.1 BIOS Handling of Device Disable

Assume that in the following power-up sequence the DEV_OFF_N signal is driven high (or it is already disabled)

1. The PCIe is established following the GIO_PWR_GOOD.
2. BIOS recognizes that the entire I210-CS/CL should be disabled.
3. The BIOS drives the DEV_OFF_N signal to the low level.
4. As a result, the I210-CS/CL samples the DEV_OFF_N signals and enters either the device disable mode.
5. The BIOS could put the link in the Electrical IDLE state (at the other end of the PCIe link) by clearing the *Link Disable* bit in the Link Control register.
6. BIOS might start with the device enumeration procedure (the entire I210-CS/CL functions are invisible).
7. Proceed with normal operation
8. Re-enable could be done by driving high the DEV_OFF_N signal, followed later by bus enumeration.

10.7 Assembly Process Flow

Figure 10-12 shows the typical process flow for mounting packages to the PCB.

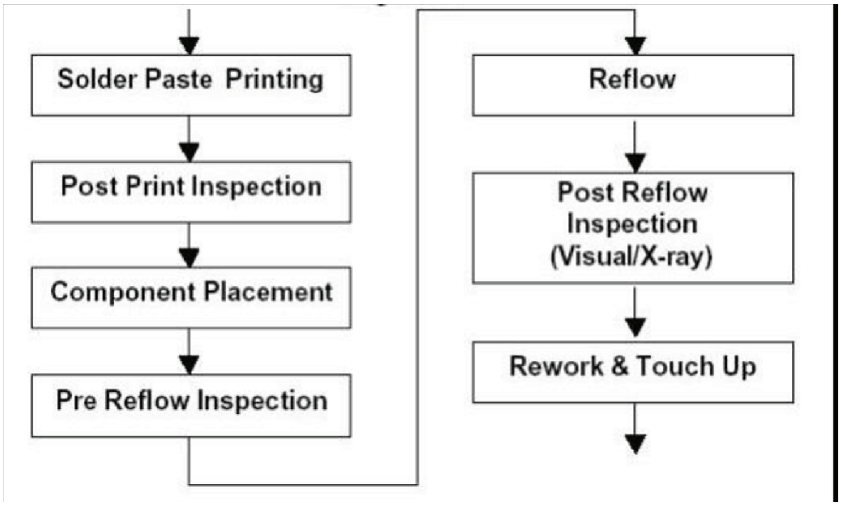


Figure 10-12.Assembly Flow

10.8 Reflow Guidelines

The typical reflow profile consists of four sections. In the preheat section, the PCB assembly should be preheated at the rate of 1 to 2 °C/sec to start the solvent evaporation and to avoid thermal shock. The assembly should then be thermally soaked for 60 to 120 seconds to remove any volatile solder paste and for activation of flux. The reflow section of the profile, the time above liquidus should be between 45 to 60 seconds with a peak temperature in the range of 245 to 250 °C, and the duration at the peak should not exceed 30 seconds. Finally, the assembly should undergo cool down in the fourth section of the profile. A typical profile band is provided in Figure 10-13, in which 220 °C is referred to as an approximation of the liquidus point. The actual profile parameters depend upon the solder paste used and specific recommendations from the solder paste manufacturers should be followed.

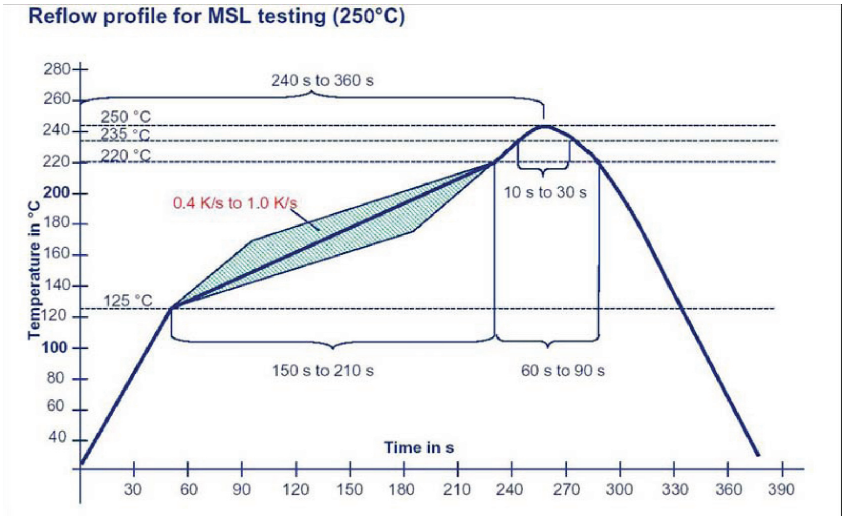


Figure 10-13.Typical Profile Band

Note:

1. Preheat: 125 °C -220 °C, 150 - 210 s at 0.4 k/s to 1.0 k/s
2. Time at T > 220 °C: 60 - 90 s
3. Peak Temperature: 245-250 °C
4. Peak time: 10 - 30 s
5. Cooling rate: <= 6 k/s
6. Time from 25 °C to Peak: 240 – 360 s

10.9 XOR Testing

A common board or system-level manufacturing test for proper electrical continuity between the I210-CS/CL and the board is some type of cascaded-XOR or NAND tree test. The I210-CS/CL implements an XOR tree spanning most I/O signals. The component XOR tree consists of a series of cascaded XOR logic gates, each stage feeding in the electrical value from a unique pin. The output of the final stage of the tree is visible on an output pin from the component.

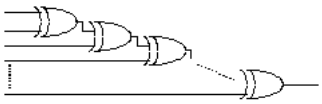


Figure 10-14.XOR Tree Concept

By connecting to a set of test-points or bed-of-nails fixture, a manufacturing test fixture can test connectivity to each of the component pins included in the tree by sequentially testing each pin, testing each pin when driven both high and low, and observing the output of the tree for the expected signal value and/or change.

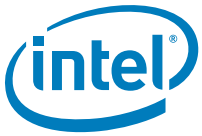
Note: Some of the pins that are inputs for the XOR test are listed as "may be left disconnected" in the pin descriptions. If XOR test is used, all inputs to the XOR tree must be connected.

When the XOR tree test is selected, the following behaviors occur:

- Output drivers for the pins listed as "tested" are all placed in high-impedance (tri-state) state to ensure that board/system test fixture can drive the tested inputs without contention.
- Internal pull-up and pull-down devices for pins listed as "tested" are also disabled to further ensure no contention with the board/system test fixture.
- The XOR tree is output on the LED1 pin.

To enter the XOR tree mode, a specific JTAG pattern must be sent to the test interface. This pattern is described by the following TDF pattern: (dh = Drive High, dl = Drive Low)

```
dh (JTAG_TDI) dl(JTAG_TCK,JTAG_TMS);

dh(JTAG_TCK);
dl(JTAG_TCK);

dh(JTAG_TMS);

loop 2
dh(JTAG_TCK);
dl(JTAG_TCK);
end loop

dl(JTAG_TMS);

loop 2
dh(JTAG_TCK);
dl(JTAG_TCK);
end loop
```



```
dl(JTAG_TDI);
dh(JTAG_TCK);
dl(JTAG_TCK);

dh(JTAG_TDI);
dh(JTAG_TCK);
dl(JTAG_TCK);

dl(JTAG_TDI);
dh(JTAG_TCK);
dl(JTAG_TCK);

dh(JTAG_TDI);
dh(JTAG_TCK);
dl(JTAG_TCK);

dl(JTAG_TDI);
dh(JTAG_TCK);
dl(JTAG_TCK);

dh(JTAG_TDI)
dh(JTAG_TMS);
dh(JTAG_TCK);
dl(JTAG_TCK);

dl(JTAG_TMS);
dh(JTAG_TCK);
dl(JTAG_TCK);

dh(JTAG_TMS);
dh(JTAG_TCK);
dl(JTAG_TCK);
dh(JTAG_TCK);
dl(JTAG_TCK);

dl(JTAG_TMS);
dh(JTAG_TCK);
dl(JTAG_TCK);

hold(JTAG_TMS,JTAG_TCK,JTAG_TDI);
```

Note: XOR tree reads left-to-right top-to-bottom.

Table 10-16. I210-CS/CL Tested Pins Included in XOR Tree (6 pins)

Pin Name	Pin Name	Pin Name
LED2	LED0	LED1 (output of the XOR tree)
NVM_SK	NVM_SO	NVM_CS_N

11.0 Thermal Considerations

This section helps design a thermal solution for systems implementing the I210-CS/CL. It details the maximum allowable operating junction and case temperatures and provides the methodology necessary to measure these values. It also outlines the results of thermal simulations of the I210-CS/CL in a standard JEDEC test environment with a 2s2p board using various thermal solutions.

11.1 Intended Audience

The intended audience for this section is system design engineers using the I210-CS/CL. System designers are required to address component and system-level thermal challenges as the market continues to adopt products with higher speeds and port densities. New designs might be required to provide more effective cooling solutions for silicon devices depending on the type of system and target operating environment

11.2 Considerations

In a system environment, the temperature of a component is a function of both the system and component thermal characteristics. System-level thermal constraints consist of the local ambient temperature at the component, the airflow over the component and surrounding board, and the physical constraints at, above, and surrounding the component that might limit the size of a thermal solution.

The component's case and die temperature are the result of:

- Component power dissipation
- Component size
- Component packaging materials
- Type of interconnection to the substrate and motherboard
- Presence of a thermal cooling solution
- Power density of the substrate, nearby components, and motherboard

All of these parameters are pushed by the continued trend of technology to increase performance levels (higher operating speeds, MHz) and power density (more transistors). As operating frequencies increase and package size decreases, the power density increases and the thermal cooling solution space and airflow become more constrained. The result is an increased emphasis on optimizing system design to ensure that thermal design requirements are met for each component in the system.

11.3 Thermal Management Importance

The objective of thermal management is to ensure that all system component temperatures are maintained within their functional limits. The functional temperature limit is the range in which the electrical circuits are expected to meet specified performance requirements. Operation outside the functional limit can degrade system performance, cause logic errors, or cause device and/or system damage. Temperatures exceeding the maximum operating limits can result in irreversible changes in the device operating characteristics. Also note that sustained operation at a component maximum temperature limit can affect long-term device reliability.

11.4 Terminology and Definitions

The following is a list of the terminology that is used in this section and their definitions:

QFN: Quad Flatpack No leads: A surface-mount package using a QFN structure whose PCB-interconnect method consists of Pb-free perimeter lands and an exposed thermal pad on the interconnect side of the package that are attached to a near chip-scale size substrate.

2s2p: A 4-layer board with two signal layers on the outside and two internal plane layers.

Thermal Resistance: The resulting change in temperature per watt of heat that passes from one reference point to another.

Junction: Refers to a P-N (diode) junction on the silicon. In this document, it is used as a temperature reference point (for example, ΘJA refers to the "junction" to "ambient" thermal resistance).

Ambient: Refers to the local ambient temperature of the bulk air approaching the component. It can be measured by placing a thermocouple approximately 1 inch upstream from the component edge.

Lands: The pads on the PCB to which BGA balls are soldered.

PCB: Printed circuit board.

Printed Circuit Assembly (PCA): A PCB that has components assembled on it.

Thermal Design Power (TDP): The estimated maximum possible/expected power generated in a component by a realistic application. TDP is a system design target associated with the maximum component operating temperature specifications. Maximum power values are determined based on typical DC electrical specification and maximum ambient temperature for a worst-case realistic application running at maximum utilization.

LFM: A measure of airflow velocity in Linear Feet per Minute.

Θ_{JA} (Theta JA): Thermal resistance from component junction to ambient, °C/W.

Ψ_{JT} (Psi JT): Junction-to-top (of package) thermal characterization parameter, °C/W. Ψ_{JT} does not represent thermal resistance, but instead is a characteristic parameter that can be used to convert between T_j and T_{case} when knowing the total TDP. Ψ_{JT} is easy to characterize in simulations or measurements and is defined as follows: $\Psi_{JT} = \frac{T_J - T_{Case}}{TDP}$ This parameter can vary with environmental conditions, such as airflow, thermal solution presence, and design

11.5 Package Thermal/Mechanical Specifications and Limit

11.5.1 Thermal Limits - Max Junction/Case

To ensure proper operation of the I210-CS/CL, the thermal solution must dissipate the heat generated by the component and maintain a case temperature at or below the values listed in Table 11-17.

The I210-CS/CL is designed to operate properly as long as the T_{case} rating is not exceeded. Section 11.7.1 discusses proper guidelines for measuring the case temperature.

Table 11-17. Absolute Maximum Case Temperature

Measured TDP (W)	$T_{case-max}$ (°C)
0.74 W @ 70 °C Ambient Temperature	85
0.80 W @ 85 °C Ambient Temperature	105

The thermal limits listed in Table 11-17 are based on simulated results of the package assembled on a standard multi-layer, 2s2p board with 1oz internal planes and 2oz external trace layers in a forced convection environment. The maximum case temperature is based on the maximum junction temperature and defined by the relationship, $T_{case-max} = T_{j-max} - (\Psi_{JT} * P_{TDP})$ where Ψ_{JT} is the junction-to-top (of package) thermal characterization parameter. If the case temperature exceeds the specified T_{case} max, thermal enhancements such as heat sinks or forced air is required.

Analysis indicates that real applications are unlikely to cause the I210-CS/CL to be at $T_{case-max}$ for sustained periods of time, given a properly designed thermal solution. Sustained operation at $T_{case-max}$ might affect long-term reliability of the I210-CS/CL and the system and thus should be avoided.

11.5.2 Thermal Specifications

The following table lists the package specific parameters under different conditions and environments. The values Θ_{JA} and Ψ_{JT} should be used as reference only as they will vary by system environment and thermal solution. Unless otherwise noted, the simulations were run in a JEDEC environment with a four layer (2s2p), 76.2 mm x 114.3 mm board with no heat sink.

Parameter	Equation	Conditions	No Heat Sink (°C/W)
Θ_{JA}	P = TDP	No Airflow	30.7
		1 m/s	20.9
		2 m/s	19.1
		3 m/s	18.2
Ψ_{JT}	P = TDP	No Airflow	0.06
		1 m/s	0.31
		2 m/s	0.48
		3 m/s	0.63

11.5.3 Simulation Setup

A simulation environment conforming to the JEDEC JESD51-2 standard was developed using a 101.5 mm x 114.5 mm, 2s2p board according to JEDEC JESD 51-9. Simulations were run with different combinations of ambient temperature and airflow speed one solution scenario, as follows:

- No heat sink

Note: Keep the following in mind when reviewing the data that is included in this section:

- All data is preliminary and is not validated against physical samples.
- Your system design might be significantly different.
- A larger board with more than four copper layers might improve the I210-CS/CL thermal performance.

11.6 Simulation Results

Table 1 lists the T_{case} as a function of airflow and ambient temperature with the component operating at the Thermal Design Power (TDP) in the environment previously listed. This table can be used as an aid in determining a starting point for the optimum airflow for the I210-CS/CL.

Again, your system design might vary considerably from the environment used to generate these values.

Note: Thermal models are available upon request (Flotherm: Detailed Model). Contact your local Intel sales representative for the I210-CS/CL thermal models.

Table 11-18. Thermal Simulation Results for Various Environmental Conditions

Ambient Temperature (°C)

T_C	0	50	100	150	200	250	300	350	400	
45	68.49	65.06	65.06	65.06	65.06	65.06	65.06	65.06	65.06	
50	73.33	69.98	69.24	69.24	69.24	69.24	69.24	69.24	69.24	
55	78.16	74.89	74.18	73.74	73.42	73.17	72.97	72.79	72.64	
60	83	79.82	79.13	78.7	78.38	78.14	77.94	77.77	77.61	0.74W
65	87.74	84.74	84.07	83.65	83.35	83.11	82.91	82.74	82.59	
70	92.68	89.66	89.02	88.61	88.31	88.08	87.88	87.71	87.56	
75	99.3	96.17	95.5	95.07	94.76	94.51	94.3	94.12	93.96	
80	104.1	101.1	100.4	100	99.72	99.47	99.27	99.09	98.93	0.80W
85	109	106	105.4	105	104.7	104.4	104.2	104.1	103.9	

No Heat Sink

Note: The red value(s) indicate airflow/ambient combinations that exceed the allowable case temperature.

11.7 Component Measurement Methodology

Measurement methodologies for determining the case and junction temperature are outlined in the sections that follow.

11.7.1 Case Temperature Measurements

Special care is required when measuring the T_{case} temperature to ensure an accurate temperature measurement is produced. Use the following guidelines when measuring T_{case}:

- Use 36-gauge (maximum) K-type thermocouples.
- Calibrate the thermocouple before making temperature measurements.
- Measure the surface temperature of the case in the geometric center of the case top.

Note: It is critical that the thermocouple bead be completely in contact with the package surface.

- Use thermally conductive epoxies, as necessary (again, ensuring the thermocouple bead is in contact with the package surface).

Care must be taken in order to avoid introducing error into the measurements when measuring a surface temperature. Measurement error might be induced by:

- Poor thermal contact between the thermocouple junction and the surface of the package.
- Contact between the thermocouple cement and the heat-sink base (if used).
- Heat loss through thermocouple leads.

11.7.1.1 Attaching the Thermocouple (No Heat Sink)

Following the guidelines listed, attach the thermocouple at a 0° angle if there is no interference with the thermocouple attach location or leads (see Figure 2).

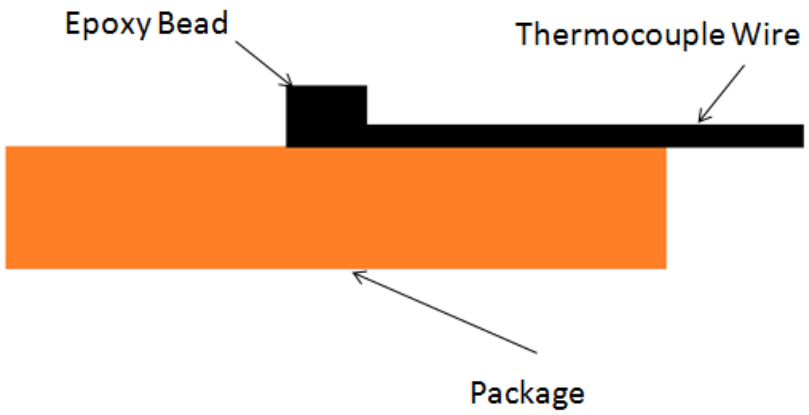
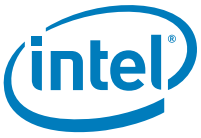


Figure 11-15.Technique for Measuring T_{case} with 0° Angle Attachment, No Heat Sink



11.8 PCB Layout Guidelines

The following general PCB design guidelines are recommended to maximize the thermal performance of QFN packages:

- When connecting ground (thermal) vias to the ground planes, do not use thermal-relief patterns.
- Thermal-relief patterns are designed to limit heat transfer between the vias and the copper planes, thus constricting the heat flow path from the component to the ground planes in the PCB.
- As board temperature also has an effect on the thermal performance of the package, avoid placing the I210-CS/CL adjacent to high-power dissipation devices.
- If airflow exists, locate the components in the mainstream of the airflow path for maximum thermal performance. Avoid placing the components downstream, behind larger devices or devices with heat sinks that obstruct or significantly preheat the air flow.

Note: The previous information is provided as a general guideline to help maximize the thermal performance of the components.

11.9 Conclusion

Increasingly complex systems require more robust and well thought out thermal solutions. The use of system air, ducting, passive or active heat sinks, or any combination thereof can help lead to a low cost solution that meets your environmental constraints.

The simplest and most cost-effective method is to improve the inherent system cooling characteristics through careful design and placement of fans, vents, and ducts. When additional cooling is required, thermal enhancements can be implemented in conjunction with enhanced system cooling. The size of the fan or heat sink can be varied to balance size and space constraints with acoustic noise.

Use the data and methodologies in this section as a starting point when designing and validating a thermal solution for the I210-CS/CL. By maintaining the I210-CS/CL's case temperature below those recommended in this section, the I210-CS/CL functions properly and reliably.

12.0 Diagnostics

12.1 Customer Visible Features

12.1.1 JTAG Test Mode Description

The I210-CS/CL includes a JTAG (TAP) port that is compliant with the IEEE standard 1149.1, 2001 Edition (JTAG). Institute of Electrical and Electronics Engineers (IEEE).

The TAP controller is accessed serially through the four dedicated pins TCK, TMS, TDI, and TDO. TMS, TDI, and TDO operate synchronously with TCK which is independent of all other clock within the I210-CS/CL. This interface can be used for test and debug purposes. System board interconnects can be DC tested using the boundary scan logic in pads. Table 12-1 shows TAP controller related pin descriptions. Table 12-2 describes the TAP instructions supported by the I210-CS/CL. The default instruction after JTAG reset is IDCODE.

Table 12-1. TAP Controller Pins

Signal	I/O	Description
TCK	In	Test clock input for the test logic defined by IEEE1149.1. **Note:** Signal should be connected to ground through a 3.3 KΩ pull-down resistor.
TDI	In	Test Data Input. Serial test instructions and data are received by the test logic at this pin. **Note:** Signal should be connected to VCC33 through a 3.3 KΩ pull-up resistor.
TDO	O/D	Test Data Output. The serial output for the test instructions and data from the test logic defined in IEEE1149.1. **Note:** Signal should be connected to VCC33 through a 3.3 KΩ pull-up resistor.
TMS	In	Test Mode Select input. The signal received at TMS is decoded by the TAP controller to control test operations. **Note:** Signal should be connected to VCC33 through a 3.3 KΩ pull-up resistor.

Table 12-2. TAP Instructions Supported

Instruction	Description	Comment
BYPASS	The BYPASS command selects the Bypass Register, a single bit register connected between TDI and TDO pins. This allows more rapid movement of test data to and from other components in the system.	IEEE 1149.1 Std. Instruction
EXTEST	The EXTEST Instruction allows circuitry or wiring external to the devices to be tested. Boundary-scan Register Cells at outputs are used to apply stimulus while Boundary-scan cells at input pins are used to capture data.	IEEE 1149.1 Std. Instruction
SAMPLE / PRELOAD	The SAMPLE/PRELOAD instruction is used to allow scanning of the boundary scan register without causing interference to the normal operation of the device. Two functions can be performed by use of the Sample/Preload instruction. SAMPLE – allows a snapshot of the data flowing into and out of a device to be taken without affecting the normal operation of the device. PRELOAD – allows an initial pattern to be placed into the boundary scan register cells. This allows initial known data to be present prior to the selection of another boundary-scan test operation.	IEEE 1149.1 Std. Instruction
IDCODE	The IDCODE instruction is forced into the parallel output latches of the instruction register during the Test-Logic-Reset TAP state. This allows the device identification register to be selected by manipulation of the broadcast TMS and TCK signals for testing purposes, as well as by a conventional instruction register scan operation. The ID code value for all the I210-CS/CL A0 SKUs is 0x01531013 (Intel's Vendor ID = 0x13, Device ID = 0x1531, Rev ID = 0x0). The ID code value for all the I210-CS/CL A1 SKUs is 0x11531013 (Intel's Vendor ID = 0x13, Device ID = 0x1531, Rev ID = 0x1). The ID code value for all the I210-CS/CL A2 SKUs is 0x31531013 (Intel's Vendor ID = 0x13, Device ID = 0x1531, Rev ID = 0x3).	IEEE 1149.1 Std. Instruction
USERCODE	After device reset and before the Device ID is read from Flash: For the I210-CS/CL A0 it is 0x01531013 (Intel's Vendor ID = 0x13, Device ID = 0x1531, Rev ID = 0x0). For the I210-CS/CL A1 it is 0x11531013 (Intel's Vendor ID = 0x13, Device ID = 0x1531, Rev ID = 0x1). For the I210-CS/CL A2 it is 0x31531013 (Intel's Vendor ID = 0x13, Device ID = 0x1531, Rev ID = 0x3). Once the Flash is read, the 16 Device ID bits, which are embedded in the USER code value reflect the device SKU.	IEEE 1149.1 Std. Instruction
HIGHZ	The HIGHZ instruction is used to force all outputs of the device (except TDO) into a high impedance state. This instruction shall select the Bypass Register to be connected between TDI and TDO in the Shift-DR controller state.	IEEE 1149.1 Std. Instruction

Appendix A. Packet Types

This section describes the packet types supported by the header split/replication and other features.

A.1 Packet Types for Header Split/Replication

The following packet types describe the different formats of the packets that are supported by the packet split or replicate feature in the I210-CS/CL. It describes the packets in the split-header point of view. This means that when describing the different fields that are checked and compared, the Header Split/Replication feature emphasizes only the fields that are needed to calculate the header length. This section describes the checks that are done after the decision to pass the packet to the host memory was made.

A.1.1 Terminology

- Compare - The field values are compared to the values that are specified in this section. For a positive result to the compare the values must be equal.
- Checked - The value of the field is compared to the recalculated value (header length …), as opposed to values specified here.
- Ignore - The field value is ignored but the field is counted to be part of the header.

A.1.2 Type 0 Ethernet (VLAN/SNAP)

This packet type contains an Ethernet header. If only *PSRTYPE.PSR_TYPE0* bit is set, the packet is split at the Ethernet header, even if additional headers are present. If other types are set, the header buffer might contain higher level headers.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type		Ignore	IP

A.1.3 Type 1 Ethernet (VLAN/SNAP) IP Packets

A.1.3.1 Type 1.1 Ethernet, IP, Data

This packet type contains only Ethernet and IPv4 headers while the payload header of the IP is not IPv6/TCP/UDP. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE1* is set.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x0800	Compare	IP
IPv4 Header					
14+D+S	1	Version/ HDR length	0x4X	Compare	Check IPv4 and header length
15+D+S	1	Type of Service	-	Ignore	
16+D+S	2	Packet Length	-	Ignore	
18+D+S	2	Identification	-	Ignore	
20+D+S	2	Fragment Info	>0 or MF bit is set	Check	Check that the packet is fragmented
22+D+S	1	Time to live	-	Ignore	
23+D+S	1	Protocol		Ignore	Has no meaning if the packet is fragmented
24+D+S	2	Header Checksum	-	Ignore	
26+D+S	4	Source IP Address	-	Ignore	
30+D+S	4	Destination IP Address	-	Ignore	
34+D+S	N	Possible IP Options		Ignore	

In this case the packet will be cut after (34+D+S+N) bytes.

- The header of the packet is split only if the packet is a fragmented packet.

  N = (IP HDR length – 5) * 4

A.1.3.2 Type 1.2: Ethernet (VLAN/snap), IPv4, TCP

This packet type contains all three Ethernet, IPv4, and TCP headers. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE2* is set

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x0800	Compare	IP
IPv4 Header					
14+D+S	1	Version/ HDR length	0x4X	Compare	Check IPv4 and header length
15+D+S	1	Type of Service	-	Ignore	
16+D+S	2	Packet Length	-	Ignore	
18+D+S	2	Identification	-	Ignore	
20+D+S	2	Fragment Info	0x00	Compare	
22+D+S	1	Time to live	-	Ignore	
23+D+S	1	Protocol	0x06	Compare	TCP header
24+D+S	2	Header Checksum	-	Ignore	
26+D+S	4	Source IP Address	-	Ignore	
30+D+S	4	Destination IP Address	-	Ignore	
34+D+S	N	Possible IP Options		Ignore	
TCP Header					
34+D+S+N	2	Source Port	Not (0x801)	Check	Not NFS packet
36+D+S+N	2	Destination Port	Not (0x801)	Check	Not NFS packet
38+D+S+N	4	Sequence number	-	Ignore	
42+D+S+N	4	Acknowledge number	-	Ignore	
46+D+S+N	1/2	Header Length		Check	
46.5+D+S+N	1.5	Different bits	-	Ignore	
48+D+S+N	2	Window size	-	Ignore	
50+D+S+N	2	TCP checksum	-	Ignore	
52+D+S+N	2	Urgent pointer	-	Ignore	
54+D+S+N	F	TCP options	-	Ignore	

In this case the packet is split after (54+D+S+N+F) bytes.

N = (IP HDR length –5) * 4.
F = (TCP header length – 5) * 4.

A.1.3.3 Type 1.3: Ethernet (SNAP/VLAN), IPv4, UDP

This packet type contains all three Ethernet, IPv4, and UDP headers. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE3* is set.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x0800	Compare	IP
IP Header					
14+D+S	1	Version/ HDR length	0x4X	Compare	Check IPv4 and header length
15+D+S	1	Type of Service	-	Ignore	
16+D+S	2	Packet Length	-	Ignore	
18+D+S	2	Identification	-	Ignore	
20+D+S	2	Fragment Info	(xx00) 000h	Compare	
22+D+S	1	Time to live	-	Ignore	
23+D+S	1	Protocol	0x11	Compare	UDP header
24+D+S	2	Header Checksum	-	Ignore	
26+D+S	4	Source IP Address	-	Ignore	
30+D+S	4	Destination IP Address	-	Ignore	
34+D+S	N	Possible IP Options		Ignore	
UDP Header					
34+D+S+N	2	Source Port	Not (0x801)	Check	Not NFS packet
36+D+S+N	2	Destination Port	Not (0x801)	Check	Not NFS packet
38+D+S+N	2	Length	-	Ignore	
40+D+S+N	2	Checksum	-	Ignore	

In this case the packet is split after (42+D+S+N) bytes.

A.1.3.4 Type 1.4: Ethernet, IPv4, IPv6

A.1.3.4.1 Ipv6 Header Options Processing

If the next header field in the IPv6 header is equal to 0x00/0x2B/0x2C/0x3B/0x3c then the next header is an IPv6 option header with the following structure:

Next Header (8 bit)	Header Len (8 bit	
Option Header Parameters		

Header Len determines the length of the header while the next header field determines the identity of the next header (could be any IPv6 extension header or another IPv6 header option).

A.1.3.4.2 The header of this type of packet is split/replicated only if PSRTYPE.PSR_TYPE is set.

A.1.3.4.3 IPv6 Next Header Values

When parsing an IPv6 header, the I210-CS/CL does not parse every kind of extension header. Packets containing an extension header that are not supported by the I210-CS/CL is treated as an unknown payload after the IPv6 header. The next header in a fragment header is ignored and this extension header is expected to be the last header.

A.1.3.4.4 Type 1.4.1: Ethernet (VLAN/SNAP), IPv4, IPv6, data

Value	Header Type
0x00	Hop by Hop
0x2B	Routing
0x2C	Fragment
0x3B	No next header (EOL)
0x3C	Destination option header

This packet type contains all three Ethernet, IPv4, and IPv6 headers. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE4* is set.

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x0800	Compare	IP
IPv4 Header					
14+D+S	1	Version/ DR length	0x4X	Compare	Check IPv4 and header length
15+D+S	1	Type of Service	-	Ignore	
16+D+S	2	Packet Length	-	Ignore	

Offset	# of Bytes	Field	Value	Action	Comment
18+D+S	2	Identification	-	Ignore	
20+D+S	2	Fragment Info	0x00	Compare	
22+D+S	1	Time to live	-	Ignore	
23+D+S	1	Protocol	0x29	Compare	Ipv6
24+D+S	2	Header Checksum	-	Ignore	
26+D+S	4	Source IP Address	-	Ignore	
30+D+S	4	Destination IP Address	-	Ignore	
34+D+S	N	Possible IP Options		Ignore	
Ipv6 Header					
34+D+S+N	1	Version/ Traffic Class	0x6X	Compare	Check IPv6
35+D+S+N	3	Traffic Class/Flow Label	-	Ignore	
38+D+S+N	2	Payload Length	-	Ignore	
40+D+S+N	1	Next Header	IPv6 extension headers	Check	
41+D+S+N	1	Hop Limit	-	Ignore	
42+D+S+N	16	Source Address	-	Ignore	
48+D+S+N	16	Destination Address		Ignore	
74+D+S+N	B	Possible IPv6 Next Headers	-	Ignore	

In this case the packet is split after (74+D+S+N+B) bytes.

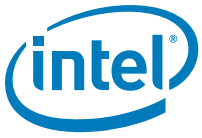
N = (IP HDR length – 5) * 4.

One of the extension headers of the IPv6 packets must be a fragment header in order for the packet to be parsed.

A.1.3.4.5 Type 1.4.2: Ethernet (VLAN/SNAP), IPv4, IPv6, TCP

This packet type contains all four Ethernet, IPv4, IPv6, and TCP headers. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE5* is set.

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x0800	Compare	IP
IPv4 Header					



Offset	# of Bytes	Field	Value	Action	Comment
14+D+S	1	Version/ HDR length	0x4X	Compare	Check IPv4 and header length
15+D+S	1	Type of Service	-	Ignore	
16+D+S	2	Packet Length	-	Ignore	
18+D+S	2	Identification	-	Ignore	
20+D+S	2	Fragment Info	0x00	Compare	
22+D+S	1	Time to live	-	Ignore	
23+D+S	1	Protocol	0x29	Compare	Ipv6
24+D+S	2	Header Checksum	-	Ignore	
26+D+S	4	Source IP Address	-	Ignore	
30+D+S	4	Destination IP Address	-	Ignore	
34+D+S	N	Possible IP Options		Ignore	
Ipv6 Header					
34+D+S+N	1	Version/ Traffic Class	0x6X	Compare	Check IPv6
35+D+S+N	3	Traffic Class/Flow Label	-	Ignore	
38+D+S+N	2	Payload Length	-	Ignore	
40+D+S+N	1	Next Header	Ipv6 extension header Or 0x06 (TCP)	Check	IPv6 extension headers
41+D+S+N	1	Hop Limit	-	Ignore	
42+D+S+N	16	Source Address	-	Ignore	
58+D+S+N	16	Destination Address		Ignore	
74+D+S+N	B	Possible IPv6 Next Headers	-	Ignore	
TCP Header					
74+T	2	Source Port	Not (0x801)	Check	Not NFS packet
76+T	2	Destination Port	Not (0x801)	Check	Not NFS packet
78+T	4	Sequence number	-	Ignore	
82+T	4	Acknowledge number	-	Ignore	
86+T	1/2	Header Length		Check	
86.5+T	1.5	Different bits	-	Ignore	
88+T	2	Window size	-	Ignore	
90+T	2	TCP checksum	-	Ignore	
92+T	2	Urgent pointer	-	Ignore	
94+T	F	TCP options	-	Ignore	

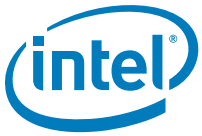
In this case the packet is split after (94+D+S+N+B+F) bytes.

T = D+S+N+B
N = (IP HDR length – 5) * 4.
F = (TCP HDR length – 5)*4

A.1.3.4.6 Type 1.4.3: Ethernet (VLAN/SNAP), IPv4, IPv6, UDP

This packet type contains all four Ethernet, IPv4, IPv6, and UDP headers. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE6* is set.

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x0800	Compare	IP
IPv4 Header					
14+D+S	1	Version/ HDR length	0x4X	Compare	Check IPv4 and header length
15+D+S	1	Type of Service	-	Ignore	
16+D+S	2	Packet Length	-	Ignore	
18+D+S	2	Identification	-	Ignore	
20+D+S	2	Fragment Info	0x00	Compare	
22+D+S	1	Time to live	-	Ignore	
23+D+S	1	Protocol	0x29	Compare	Ipv6
24+D+S	2	Header Checksum	-	Ignore	
26+D+S	4	Source IP Address	-	Ignore	
30+D+S	4	Destination IP Address	-	Ignore	
34+D+S	N	Possible IP Options		Ignore	
Ipv6 Header					
34+D+S+N	1	Version/ Traffic Class	0x6X	Compare	Check IPv6
35+D+S+N	3	Traffic Class/Flow Label	-	Ignore	
38+D+S+N	2	Payload Length	-	Ignore	
40+D+S+N	1	Next Header	IPv6 extension header or 0x11 (UDP)	Check	IPv6 extension headers:
41+D+S+N	1	Hop Limit	-	Ignore	
42+D+S+N	16	Source Address	-	Ignore	
58+D+S+N	16	Destination Address		Ignore	
74+D+S+N	B	Possible IPv6 Next Headers	-	Ignore	



Offset	# of Bytes	Field	Value	Action	Comment
UDP Header					
74+D+S+N+B	2	Source Port	Not (0x801)	Check	Not NFS packet
76+D+S+N+B	2	Destination Port	Not (0x801)	Check	Not NFS packet
78+D+S+N+B	2	Length	-	Ignore	
80+D+S+N+B	2	Checksum	-	Ignore	

In this case the packet is split after (82+D+S+N+B) bytes.

N = (IP HDR length – 5) * 4.

A.1.4 Type 2: Ethernet, IPv6

A.1.4.1 Type 2.1: Ethernet (VLAN/SNAP), IPv6, Data

This packet type contains both Ethernet and IPv6 headers while the packet should be a fragmented packet. If the packet is not fragmented and the next header is not one of the supported types from Section A.1.3.4 then the header is not split. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE7* is set.

Offset	# of Bytes	Field	Value (hex)	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA03 0000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x86DD	Compare	IP
IPv6 Header					
14+D+S	1	Version/ Traffic Class	0x6X	Compare	Check IPv6
15+D+S	3	Traffic Class/Flow Label	-	Ignore	
18+D+S	2	Payload Length	-	Ignore	
20+D+S	1	Next Header	IPv6 next header types	Check	The last header must be fragmented header in order for the header to be split.
21+D+S	1	Hop Limit	-	Ignore	
22+D+S	16	Source Address	-	Ignore	
38+D+S	16	Destination Address		Ignore	
54+D+S	N	Possible IPv6 Next Headers	-	Ignore	

In this case the packet is split after (54+D+S+N) bytes.

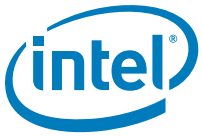
The last next header field of the IP section field should not be 0x11/0x06 (TCP/UDP).

A.1.4.2 Type 2.2: Ethernet (VLAN/SNAP) IPv6 TCP

This packet type contains all three Ethernet, IPv6, and TCP headers. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE8* is set.

Offset	# of Bytes	Field	Value (hex)	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030 000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x86DD	Compare	IP
IPv6 Header					
14+D+S	1	Version/ Traffic Class	0x6X	Compare	Check IPv6
15+D+S	3	Traffic Class/Flow Label	-	Ignore	
18+D+S	2	Payload Length	-	Ignore	
20+D+S	1	Next Header	IPv6 next header types or 0x06 (TCP)	Check	
21+D+S	1	Hop Limit	-	Ignore	
22+D+S	16	Source Address	-	Ignore	
38+D+S	16	Destination Address		Ignore	
54+D+S	N	Possible IPv6 Next Headers	-	Ignore	
TCP Header					
54+D+S+N	2	Source Port	Not (0x801)	Check	Not NFS packet
56+D+S+N	2	Destination Port	Not (0x801)	Check	Not NFS packet
58+D+S+N	4	Sequence number	-	Ignore	
62+D+S+N	4	Acknowledge number	-	Ignore	
66+D+S+N	1/2	Header Length		Check	
66.5+D+S+N	1.5	Different bits	-	Ignore	
68+D+S+N	2	Window size	-	Ignore	
70+D+S+N	2	TCP checksum	-	Ignore	
72+D+S+N	2	Urgent pointer	-	Ignore	
74+D+S+N	F	TCP options	-	Ignore	

In this case the packet is split after (54+D+S+N+F) bytes.



F = (TCP header length – 5) * 4

The last Next-header field of the last header of the IP section must be 0x11.

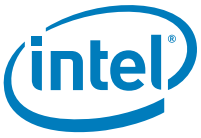
A.1.4.3 Type 2.3: Ethernet (VLAN/SNAP) IPv6 UDP

This packet type contains all three Ethernet, IPv6, and UDP headers. The header of this type of packet is split/replicated only if *PSRTYPE.PSR_TYPE9* is set.

Offset	# of Bytes	Field	Value (hex)	Action	Comment
0	6	Destination Address		Ignore	MAC Header – processed by main address filter, or broadcast
6	6	Source Address		Ignore	
12	S=(0/4/8)	Possible VLAN Tags (single or double)	0x8100 ****	Compare on internal VLAN only	
12+S	D=(0/8)	Possible LLC/SNAP Header	Length + 0xAAAA030 000	Compare	Length means a value smaller than 0x600.
12+D+S	2	Type	0x86DD	Compare	IP
IPv6 Header					
14+D+S	1	Version/ Traffic Class	0x6X	Compare	Check IPv6
15+D+S	3	Traffic Class/Flow Label	-	Ignore	
18+D+S	2	Payload Length	-	Ignore	
20+D+S	1	Next Header	IPv6 next header types Or 0x11 (UDP)	Check	
21+D+S	1	Hop Limit	-	Ignore	
22+D+S	16	Source Address	-	Ignore	
38+D+S	16	Destination Address		Ignore	
54+D+S	N	Possible IPv6 Next Headers	-	Ignore	
UDP Header					
54+D+S+N	2	Source Port	Not (0x801)	Check	Not NFS packet
56+D+S+N	2	Destination Port	Not (0x801)	Check	Not NFS packet
58+D+S+N	2	Length	-	Ignore	
60+D+S+N	2	Checksum	-	Ignore	

In this case the packet is split after (62+D+S+N) bytes.

The last Next-header field of the last header of the IP section must be 0x06.



A.1.5 Type 3: NFS Packets

NFS headers can come in all the frames that contain a UDP/TCP header. The NFS (and RPC headers) are extensions to these types of packets. All of the packets previously described in sections A.1.3.2, A.1.3.2, A.1.3.4.5, A.1.3.4.5, A.1.4.3, and A.1.4.2, can accommodate NFS headers.

PSRTYPE.PSR_TYPE11/12/14/15/18/19 controls the split/replication behavior of NFS packets. See Section 7.10.3 for details.

In this section, only the NFS (and RPC) header is described. The length of this header should be added to the length of the primary type of the packet.

The I210-CS/CL starts looking within the UDP/TCP payload to check whether it contains an NFS header. This is determined when either the source or destination port of the TCP/UDP is equal to 0x801.

- Destination port equal 0x801 => NFS write request (as received by the NFS server).
- Source port equal 0x801 => NFS read response (as received by the NFS client).

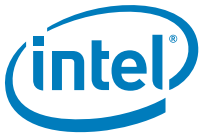
The VSZ/CSZ fields are each 4 bytes long but their actual values are less than 2 words by definition so hardware only checks the lower 2 bytes of these size fields.

RPC read requests are not described in this document since they contain only headers and no data therefore there is no need to split them.

Note: NFS over TCP is problematic – due to the fact that the RPC header might appear in the middle of the frame. It remains to be checked if software always supports putting the RPC right next to the UDP/TCP header.

A.1.5.1 Type 3.1: NFS Write Request

In all write requests, the destination port of the TCP/UDP header must be 0x801.



A.1.5.1.1 Type 3.1.1: NFS Write Request (NFSv2)

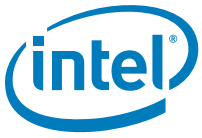
Offset	# of Bytes	Field	Value (hex)	Action	Comment
	RPC Header				
0	D =(0/4)	Record Header	-	Ignore	If the previous header was TCP header than this field contain 4 bytes
0+D	4	Message type	0x00	Compare	
4+D	4	RPC version	0x02	Compare	
8+D	4	RPC program	0x18A63	Compare	
12+D	4	Program version	0x02	Compare	
16+D	4	Procedure	0x08	Compare	
20+D	4	Credentials Size (CSZ)	<400	Check	
24+D	B	Credentials Data (CSZ)	-	Ignore	B = (CSZ pad 4)
24+D+B	4	Verifier Flavor	-	Ignore	
28+D+B	4	Verifier Size (VSZ)	<400	Check	
32+D+B	F	Verifier Data		Ignore	F = (VSZ pad 4)
NFS Header					
32+D+B+F	32	handle		Ignore	
64+D+B+F	4	begin offset		Ignore	
68+D+B+F	4	Offset		Ignore	
72+D+B+F	4	Total count		Ignore	
76+D+B_F	4	Data len		Ignore	

In this case the NFS header size is (80+D+B+F) bytes. This length should be added to the UDP/TCP type that was already parsed.

A.1.5.1.2 Type 3.1.2: NFS Write Request (NFSv3)

Offset	# of Bytes	Field	Value (hex)	Action	Comment
		RPC Header			
0	D =(0/4)	Record Header	-	Ignore	If the previous header was TCP header than this field contain 4 bytes
0+D	4	Message type	0x00	Compare	
4+D	4	RPC version	0x02	Compare	
8+D	4	RPC program	0x18A63	Compare	
12+D	4	Program version	0x03	Compare	
16+D	4	Procedure	0x07	Compare	
20+D	4	Credentials Size (CSZ)	<400	Check	
24+D	B	Credentials Data (CSZ)	-	Ignore	B = (CSZ padded to 4)
24+D+B	4	Verifier Flavor	-	Ignore	
28+D+B	4	Verifier Size (VSZ)	<400	Check	
32+D+B	F	Verifier Data		Ignore	F = (VSZ padded to 4)
		NFS Header			
32+D+B+F	4	Fhandle_size	<64	Check	
36+D+B+F	S	fhandle		Ignore	S = (Fhandle_size padded to 4)
36+D+B+F+S	8	Offset		Ignore	
44+D+B+F+S	4	Count		Ignore	
48+D+B+F+S	4	Stable_how		Ignore	
52+D+B+F+S	4	Data len		Ignore	

In this case the NFS header size is (56+D+B+F+S) bytes. This length should be added to the UDP/TCP type that was already parsed.



A.1.5.1.3 Type 3.1.3: NFS Write Request (NFSv4)

Offset	# of Bytes	Field	Value (hex)	Action	Comment
	RPC Header				
0	D =(0/4)	Record Header	-	Ignore	If the previous header was TCP header than this field contain 4 bytes
0+D	4	Message type	0x00	Compare	
4+D	4	RPC version	0x02	Compare	
8+D	4	RPC program	0x18A63	Compare	
12+D	4	Program version	0x04	Compare	
16+D	4	Procedure	0x26	Compare	
20+D	4	Credentials Size (CSZ)	<400	Check	
24+D	B	Credentials Data (CSZ)	-	Ignore	B = (CSZ pad 4)
24+D+B	4	Verifier Flavor	-	Ignore	
28+D+B	4	Verifier Size (VSZ)	<400	Check	
32+D+B	F	Verifier Data		Ignore	F = (VSZ pad 4)
NFS Header					
32+D+B+F	8	State id		Ignore	
40+D+B+F	8	Offset		Ignore	
48+D+B+F	4	Stable_how		Ignore	
52+D+B+F	4	Data len		Ignore	

In this case the NFS header size is (56+D+B+F) bytes. This length should be added to the UDP/TCP type that was already parsed.

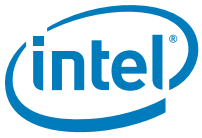
A.1.5.2 Type 3.2: NFS Read Response

A.1.5.2.1 The I210-CS/CL should be configured to the right version via its configuration space.

A.1.5.2.2 Type 3.2.1: NFS Read Response (NFSv2)

Offset	# of Bytes	Field	Value (hex)	Action	Comment
	RPC Header				
0	D =(0/4)	Record Header	-	Ignore	If the previous header was TCP header than this field contain 4 bytes
0+D	4	XID		Ignore	
4+D	4	Message type	0x01	Compare	
8+D	4	Reply status	0x00	Ignore	'0' means O.K and only if this value is '0' there will be additional data
12+D	4	Verifier Flavor	-	Ignore	
16+D	4	Verifier Size (VSZ)	<400	Check	
20+D	F	Verifier Data	-	Ignore	F = (VSZ pad 4)
20+D+F	4	Accept status	0x00	Ignore	'0' means O.K
	NFS Header				
24+D+F	4	Status	0x00	Ignore	
28+D+F	68	Attributes	-	Ignore	
96+D+F	4	Data len	-	Ignore	

In this case the NFS header size is (100+D+F) bytes. This length should be added to the UDP/TCP type that was already parsed.



A.1.5.2.3 Type 3.2.1: NFS Read Response (NFSv3)

Offset	# of Bytes	Field	Value (hex)	Action	Comment
	RPC Header				
0	D =(0/4)	Record Header	-	Ignore	If the previous header was TCP header than this field contain 4 bytes
0+D	4	XID	-	Ignore	
4+D	4	Message type	0x01	Compare	
8+D	4	Reply status	0x00	Ignore	'0' means O.K and only if this value is '0' there will be additional data
12+D	4	Verifier Flavor	-	Ignore	
16+D	4	Verifier Size (VSZ)	<400	Check	
20+D	F	Verifier Data	-	Ignore	F = (VSZ pad 4)
20+D+F	4	Accept status	0x00	Ignore	'0' means O.K
	NFS Header				
24+D+F	4	Status	0x00	Ignore	
	4	Attr_follow	-	Check	
28+D+F	S	Attributes	-	Ignore	Attr_flow=1 ? S=84: S=0
28+D+F+S	4	Count	-	Ignore	
32+D+F+S	4	Eof	-	Ignore	
36+D+F+S	4	Data len	-	Ignore	

In this case the NFS header size is (40+D+F+S) bytes. This length should be added to the UDP/TCP type that was already parsed.

A.1.5.2.4 Type 3.2.1: NFS Read Response (NFSv4)

Offset	# of Bytes	Field	Value (hex)	Action	Comment
	RPC Header				
0	D =(0/4)	Record Header	-	Ignore	If the previous header was TCP header than this field contain 4 bytes
0+D	4	XID	-	Ignore	
4+D	4	Message type	0x01	Compare	
8+D	4	Reply status	0x00	Ignore	'0' means O.K and only if this value is '0' there will be additional data
12+D	4	Verifier Flavor	-	Ignore	
16+D	4	Verifier Size (VSZ)	<400	Check	
20+D	F	Verifier Data	-	Ignore	F = (VSZ pad 4)
20+D+F	4	Accept status	0x00	Ignore	'0' means O.K
	NFS Header				
24+D+F	4	Status	0x00	Ignore	'0' means O.K
28+D+F	4	eof	-	Ignore	
32+D+F	4	Data len	-	Ignore	

In this case the NFS header size is (36+D+F) bytes. This length should be added to the UDP/TCP type that was already parsed.

A.2 IP and TCP/UDP Headers for TSO

This section outlines the format and content for the IP, TCP and UDP headers. The I210-CS/CL requires baseline information from the software device driver in order to construct the appropriate header information during the segmentation process.

Header fields that are modified by the I210-CS/CL are highlighted in the figures that follow.

Note: IPv4 requires the use of a checksum for the header. IPv6 does not use a header checksum.

IPv4 length includes the TCP and IP headers, and data. IPv6 length does not include the IPv6 header.

Note: The IP header is first shown in the traditional (such as RFC 791) representation, and because byte and bit ordering is confusing in that representation, the IP header is also shown in little endian format. The actual data is fetched from memory in little endian format.

Table A-1. IPv4 Header (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
Version / IP Hdr Length	TYPE of service	Total length	
Identification		Flags	Fragment Offset
Time to Live	Layer 4 Protocol ID	Header Checksum	

Table A-1. IPv4 Header (Traditional Representation)

Source Address
Destination Address
Options

Table A-2. IPv4 Header (Little Endian Order)

Byte3				Byte2		Byte1		Byte0	
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	
LSB	Total length				MSB	TYPE of service		Version	IP Hdr Length
Fragment Offset Low				R ES	N F	M F	Fragment Offset High	LSB	Identification MSB
Header Checksum						Layer 4 Protocol ID		Time to Live	
Source Address									
Destination Address									
Options									

Identification is incremented on each packet.

Flags Field Definition:

The Flags field is defined as follows. Note that hardware does not evaluate or change these bits.

- MF - More Fragments
- NF - No Fragments
- Reserved

The I210-CS/CL does TCP segmentation, not IP fragmentation. IP fragmentation might occur in transit through a network's infrastructure.

Table A-3. IPv6 Header (Traditional Representation)

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
Version	Priority	Flow Label				
Payload Length				Next Header Type		Hop Limit
Source Address						
Destination Address						
Extensions (if any)						

Table A-4. IPv6 Header (Little Endian Order)

Byte3				Byte2				Byte1				Byte0			
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
Flow Label																Version				Priority											
Hop Limit								Next Header Type								LSB				Payload Length											MSB
Source Address																															
Destination Address																															
Extensions																															

A TCP or UDP frame uses a 16 bit wide one's complement checksum. The checksum word is computed on the outgoing TCP or UDP header and payload, and on the pseudo header. Details on checksum computations are provided in Section 6.2.4.

Note: TCP and UDP over IPv6 requires the use of checksum, where it is optional for UDP over IPv4.

The TCP header is first shown in the traditional (such as RFC 793) representation, and because byte and bit ordering is confusing in that representation, the TCP header is also shown in little endian format. The actual data is fetched from memory in little endian format.

Table A-5. TCP Header (Traditional Representation)

0	1	2	3	4	5	6	7	8	9	1 0	1	2	3	4	5	6	7	8	9	2 0	1	2	3	4	5	6	7	8	9	3 0	1
Source Port																Destination Port															
Sequence Number																															
Acknowledgement Number																															
TCP Header Length				Reserved								URG	ACK	PSH	RST	SYN	FIN	Window													
Checksum																				Urgent Pointer											
Options																															

Table A-6. TCP Header (Little Endian)

Byte3								Byte2								Byte1								Byte0							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Destination Port																Source Port															
LSB								Sequence Number																							MSB
Acknowledgement Number																															
Window																RE S	URG	ACK	PSH	RST	SYN	FIN	TCP Header Length							Reserved	
Urgent Pointer																Checksum															
Options																															

The TCP header is always a multiple of 32-bit words. TCP options might occupy space at the end of the TCP header and are a multiple of 8 bits in length. All options are included in the checksum.

The checksum also covers a 96-bit pseudo header conceptually prefixed to the TCP Header (see Table A-7). For IPv4 packets, this pseudo header contains the IP Source Address, the IP Destination Address, the IP Protocol field, and TCP Length. Software pre-calculates the partial pseudo header sum, which includes IPv4 SA, DA and protocol types, but NOT the TCP length, and stores this value into the TCP checksum field of the packet. For both IPv4 and IPv6, hardware needs to factor in the TCP length to the software supplied pseudo header partial checksum.

Note: When calculating the TCP pseudo header, the byte ordering can be tricky. One common question is whether the Protocol ID field is added to the "lower" or "upper" byte of the 16- bit sum.

The Protocol ID field should be added to the least significant byte (LSB) of the 16-bit pseudo header sum, where the most significant byte (MSB) of the 16-bit sum is the byte that corresponds to the first checksum byte out on the wire.

The TCP Length field is the TCP Header Length including option fields plus the data length in bytes, which is calculated by hardware on a frame-by-frame basis. The TCP Length does not count the 12 bytes of the pseudo header. The TCP length of the packet is determined by hardware as:

TCP Length = min(MSS, PAYLOADLEN) + L5_LEN

The two flags that may be modified are defined as:

- PSH: receiver should pass this data to the application without delay
- FIN: sender is finished sending data

The handling of these flags is described in Section 6.2.4.

Payload is normally MSS except for the last packet where it represents the remainder of the payload.

Table A-7. TCP/UDP Pseudo Header Content for IPv4 (Traditional Representation)

IPv4 Source Address		
IPv4 Destination Address		
Zero	Layer 4 Protocol ID	TCP/UDP Length

Table A-8. TCP/UDP Pseudo Header Content for IPv6 (Traditional Representation)

IPv6 Source Address	
IPv6 Final Destination Address	
TCP/UDP Packet Length	
Zero	Next Header

Note: From RFC2460:

- If the IPv6 packet contains a Routing header, the Destination Address used in the pseudo-header is that of the final destination. At the originating node, that address will be in the last element of the Routing header; at the recipient(s), that address will be in the Destination Address field of the IPv6 header.
- The Next Header value in the pseudo-header identifies the upper-layer protocol (e.g., 6 for TCP, or 17 for UDP). It will differ from the Next Header value in the IPv6 header if there are extension headers between the IPv6 header and the upper-layer header.

- The Upper-Layer Packet Length in the pseudo-header is the length of the upper-layer header and data (e.g., TCP header plus TCP data). Some upper-layer protocols carry their own length information (e.g., the Length field in the UDP header); for such protocols, that is the length used in the pseudo- header. Other protocols (such as TCP) do not carry their own length information, in which case the length used in the pseudo-header is the Payload Length from the IPv6 header, minus the length of any extension headers present between the IPv6 header and the upper-layer header.
- Unlike IPv4, when UDP packets are originated by an IPv6 node, the UDP checksum is not optional. That is, whenever originating a UDP packet, an IPv6 node must compute a UDP checksum over the packet and the pseudo-header, and, if that computation yields a result of zero, it must be changed to hex FFFF for placement in the UDP header. IPv6 receivers must discard UDP packets containing a zero checksum, and should log the error.

A type 0 Routing header has the following format:

- Next Header - 8-bit selector.

Table A-9. IPv6 Routing Header (Traditional Representation)

Next Header	Hdr Ext Len	Routing Type "0"	Segments Left "n"
Reserved			
Address[1]			
Address[2]			
...			
Final Destination Address[n]			

  - — Identifies the type of header immediately following the Routing header.
  - — Uses the same values as the IPv4 Protocol field [RFC-1700 et seq.].
- Hdr Ext Len - 8-bit unsigned integer. Length of the Routing header in 8-octet units, not including the first 8 octets. For the Type 0 Routing header, Hdr Ext Len is equal to two times the number of addresses in the header.
- Routing Type - 0.
- Segments Left - 8-bit unsigned integer. Number of route segments remaining, i.e., number of explicitly listed intermediate nodes still to be visited before reaching the final destination. Equal to "n" at the source node.

Reserved - 32-bit reserved field. Initialized to zero for transmission; ignored on reception.

- Address[1...n] - Vector of 128-bit addresses, numbered 1 to n.

The UDP header is always 8 bytes in size with no options.

Table A-10. UDP Header (Traditional Representation)

0 1 2 3 4 5 6 7	8 9 10 1 2 3 4 5	6 7 8 9 20 1 2 3	4 5 6 7 8 9 30 1
Source Port		Destination Port	
Length		Checksum	

Table A-11. UDP Header (Little Endian Order)

Byte3								Byte2								Byte1								Byte0							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
Destination Port																Source Port															
Checksum																Length															

UDP pseudo header has the same format as the TCP pseudo header. The pseudo header conceptually prefixed to the UDP header contains the IPv4 source address, the IPv4 destination address, the IPv4 protocol field, and the UDP length (same as the TCP Length discussed above). This checksum procedure is the same as is used in TCP.

Unlike the TCP checksum, the UDP checksum is optional (for IPv4). Software must set the TXSM bit in the TCP/IP Context Transmit Descriptor to indicate that a UDP checksum should be inserted. Hardware will not overwrite the UDP checksum unless the TXSM bit is set.

NOTE: This page intentionally left blank.