

Intel® 82598 10 GbE Controller Datasheet

LAN Access Division

FEATURES

General
- Serial Flash Interface
- 4-wire SPI EEPROM Interface
- Configurable LED operation for software or OEM customization of LED displays
- Protected EEPROM space for private configuration
- Device disable capability
- Package Size - 31 x 31 mm

Networking
- Complies with the 10 Gb/s and 1 Gb/s Ethernet/802.3ap (KX/KX4) specification
- Complies with the 10 Gb/s Ethernet/802.3ae (XAUI) specification
- Complies with the 1000BASE-BX specification
- Support for jumbo frames of up to 16 kB
- Auto negotiation clause 73 for supported mode
- CX4 per 802.3ak
- Flow control support: send/receive pause frames and receive FIFO thresholds
- Statistics for management and RMON
- 802.1q VLAN Support
- TCP Segmentation Offload (TSO): up to 256 kB
- IPv6 support for IP/TCP and IP/UDP receive checksum offload
- Fragmented UDP checksum offload for packet reassembly
- Message Signaled Interrupts (MSI)
- Message Signaled Interrupts (MSI-X)
- Interrupt throttling control to limit maximum interrupt rate and improve CPU usage
- Multiple receive queues (RSS) 8 x 8 and 16 x 4
- 32 transmit queues
- Dynamic interrupt moderation
- DCA support
- TCP timer interrupts
- No snoop
- Relaxed ordering
- Support for 16 Virtual Machines Device queues (VMDq) per port

Host Interface
- PCI Express* (PCIe*) Specification v2.0 (2.5 GT/s)
- Bus width - x1, x2, x4, x8
- 64-bit address support for systems using more than four GB of physical memory

MAC Functions
- Descriptor ring management hardware for transmit and receive
- ACPI register set and power down functionality supporting D0 and D3 states
- A mechanism for delaying/reducing transmit interrupts
- Software-controlled global reset bit (resets everything except the configuration registers)
- Eight Software-Definable Pins (SDP) per port
- Four of the SDP pins can be configured as general-purpose interrupts
- Wakeup
- IPv6 wake-up filters
- Configurable flexible filter (through EEPROM)
- LAN function disable capability
- Programmable receive buffer of 512 kB, which can be subdivided to up-to-eight individual packet buffers
- Programmable transmit buffer of 320 kB, subdivided into up-to-eight individual packet buffers of 40 kB each
- Default Configuration by EEPROM for all LEDs for pre-driver functionality

Manageability
- Eight VLAN L2 filters
- 16 Flex L3 Port filters
- Four flexible TCO filters
- Four L3 address filters (IPv4)
- Advanced pass through-compatible management packet transmit/receive support
- SMBus interface to an external BMC
- NC-SI interface to an external BMC
- Four L3 address filters (IPv6)
- Four L2 address filters

Reference Number: 319282-007
Revision: 3.2
October 2010

Legal

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

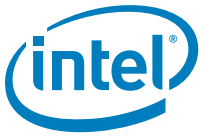
The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting Intel's Web Site.

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

Copyright © 2008, 2009, 2010; Intel Corporation. All Rights Reserved.



Revisions

Rev	Date		Comments
0.75	08/05		• First integrated version. • Added MUSIC and iSCSI TSO definition. • Initialization chapter updated • All main functions are integrated. Chapters that are not updated: • NVM memory Map. • Manageability. • Power management.
0.89	10/05		First full version. Too many changes to list here… Revision control indicates changes from Zoar.
0.9	11/05		• Pin changes to support 4 more SDP (per port), POR Baypass, Clock Bypass, power pin changes and number of Spares. • PCIe* read request size is limited to 256B • Removal of PCIe* Gen 2 support • Updated the initialization sequence for proper link setup at the different modes of network interface • Sync up with Zoar C-spec 0.94 • Many address changes in the programming interface • Ball out updated • Added Cibolo for the feature summary comparison • Changes in LAN/SAN use of RSS • Music chapter was updated, registers and statistics related to Music were updated • Added EEPROM to CSR capability • Link initialization was updated • Modified the Tx descriptors to be 8 per queue (instead of 4 and no global descriptors) • Added 10 general purpose semaphores • EEPORM PCIe* fields were re-organized, Added EEPROM words for MAC, Music • Jumbo frame support up to 16KB • Address changes in programming interface

Rev	Date		Comments
0.95	01/06		• Added the missing FC statistics • Added control bits for Music (Recycle mode, Receive LSP) • Added register A - packet types for packet split • Reference section updated • Pin names updated to mutch design • Updated Error field in the Rx descriptor • Udded a column to indicate for each register it internal block location (for internal use only) • Added PBACL, TFCS, MREVID registers • Added missing bits in Music configuration • Added missing flow control statistics (were missing in the register description) • Removed PEJC register • Added documentation to missing PCIe* registers • Fixed old refrences to RCTL to the new Oplin registers • Changed the SDP control bits location (modified registers CTRL, EXT_CTRL and added ESDP register) • Removed MPME abd MEN bits from EEPROM load and WUC, Added MEN bit for Port 1 in the EEPROM (word 0x38) • Only EEPROM section pointer of 0xFFFF will act as non valid pointer. • Removed LPE (bit 2) from FCRTL. • Added ADVD3WUC to WUC register and added an additional word to be read from the EEPROM to load this bit value. • Removed BCN control and status registers (and any other documentation) • Changed the addresses of QPTC and QBRC • Added a note that EEPROM control words are loaded after PERST, inband reast or LAN_PWR_GOOD • Removed interrupt statistics • Added Frimware EEPROM words to section 7 • Removed SOL_on and IDE_on from FWSM register (reserved) • Fixed and added registers in the MAC section to support auto negotiation, DFT and better descriptions • Added XEC stat (XSUM error) at 0x04120 • Added a new mode in DESCTYPE to always use the header buffer in split header mode • Removed the fixed partition control bits for RX/TX PB partition and added the RXPBSIZE/TXPBSIZE registers for PB partition • Removed BWG status registers • Changed the Rx status to MNG (PIF -> Reserved) • Removed SW initiated PAUSE transmission capability

Rev	Date		Comments
1.0	02/06		• Offload for Rx Ipv6 packets with "home extention" or fragments were changed back to No. • iSCSI support for Rx header split was removed (Oplin will not support this feature) • Fixed BSIZEHEADER to max at 1024 instead of 16KB as stated. • Chnaged the PCIe* analog section load time from LAN_PWR_GOOD to be after PE_RST_N instead. • Removed No MNG SMBus configuration offset from the MNG test structure pointer. • Added RCLKEXTP/N clock characterization in the electrical section • Removed the promiscuous cases that stated that PIF bit will be cleared • Made bit 25 from "RSS Field Enable" in the MRQC register reserved 0. • Added the Pullups table (section 3.2) • Removed IPIDV bit from MNG status • Changed the default value of the MNG_en bit in the GRC register so that the MNG is enabled by default • Added MNGTXMAP register that controls the mapping of the MNG transmit traffic to the appropriate TC • Added a note that For proper operation PTHRESH value should be bigger than the number of buffers needed to accommodate a single packet. • Removed Drop_En bit from RXDCTL and crerated 2 registers DROPEN0,1 that will hold all drop enable bits (bit per queue). • Changed the addresses of the RNBC statistical counters • Added a bit to bypass the descriptor monitor in the RXCTRL • Added the limitation that in Music mode Rx buffers should be x2 in BSIZEPACKET • Added a bit in RDRXCTL to reflect that the DMA init is done, this bit was also added to the SW initialization flow (to make sure software waits for DMA init done indication) • Removed line 0x13 in the packet types supported by packet split (duplicate to line 0x0) • Removed PSR_type0 as there is no L2 split • Changed the ATLASCTL register so it will support Atlas registers read • Added Atlas Tx-> Rx loopback • Added a note that under Music Jumbo frames support is restricted to 9KB. • Added a mode for TXPBSIZE to support diagnostics that enables the full Tx PB (320K) as a single packet buffer. • Added an EEPROM word that enables the loading of AUTOC2 (upper half) from the EEPROM. • Added a new statistics for Error byte count (0x04008) • Added bits in AUTOC that reflect the connection speed type (BX,KX,KX4,CX4,XAUI) • Changed the format of the "write configuration command" (BMC -> Oplin) to support 3 bytes of address instead of 2. • Chenged the format of the "read configuration request" (Oplin -> BMC) to support 3 bytes of address instead of 2. • Restricted BSIZEHEADER to 1K bytes

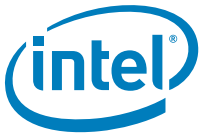
Rev	Date		Comments
1.05	03/06		• Changed the following registers from RW to RO PCS1GDBG0 (0x04210; RO) PCS1GDBG1 (0x04214; RO) PCS1GANLP (0x0421C; RO) PCS1GANLPNP (0x04224; RO) • Removed MBSDC statistic (address 0x04018) • Changed the manageability filtering diagram so it will match the text (added VLAN filtering for Host after the broadcast filtering) • Changed HTHRESH possible values to be 0 only • Added two interrupt control bits in GPIE. • Removed the TOTL/H registers from the documentation as they were mapped to the GOTL/H and Oplin will always transmit good packets • Removed PCS1GSTA2 register, the DFT capability of this register is covered by other registers. • Changed the structure of EEPROM Word SDP control - offset 6 so it will match the ESDP register structure. • Fixed "other causes" bit in the EICR to cover bots 29:20 in the EICR • Included GPIE.MSIX_MODE bit to indicate to provide the interrupt mode of operation to the interrupt block • Included DCN001 Ballout and pin changes • Included DCN003 EEPROM recovery and protection • Included DCN015 Change SPARE[7] to IDDQ_MODE_EN • Included DCN014 Support for clause 59 test pattern • Included DCN007 Removal of receive iSCSI header split support • Included DCN006 Removal of transmit iSCSI header CRC offload support • Included DCN004 MAC speed change at different power modes • Included DCN005 New BAR for MSI-X • Included DCN008 support for SAN-LAN queue mapping in Rx • Included DCN016 DMA/DBU die size reduction - statistical counters reductions • Changed the PCIe* capability version default to be 2h indicating PCIe* 2.0 • Added Configuration Structure to the MNG EEPROM section • Added "Wait for manageability configuration done indication (EEMNGCTL.CFG_DONE)" to the SW initialization sequence
1.08			• Added (back) PCIe* x2 support • E-spec was updated according per review inputs

Rev	Date		Comments
1.1			• LINK loop-back section was updated (?13.3) • A note to swizzle bits (?9.2.3.12.25) • RT2CR register was changed • The addresses of the following registers were changed: RQSMR, TQSMR, QPRC, QPTC, QBRC and QBTC. • HSMC0/1R registers were removed • GSCN_0/1/2/3 changed to read only • Change default of PCIe* max read request size to 512 (?4.1.9.7) • GHOST ECC register was added (?13.2.35) • Priority flow control and LINK flow control can not be enabled together, • EPROM control words 0x0 and 0x38 were changed (?7.3.1) • Two bits were added to GPIE register (?9.2.3.5.15) • HICR register was updated(?9.2.3.16.1.2) • Next Capability Pointer default was changed (?4.1.9.8.1.1) • RMCS and PDPMCS registers were changed • Fixed the note so that master disable bit description will indicate this bit should be cleared by reset and not by the device driver ?6.2.5.3.2, ?9.2.3.1.1) • Removed the sentence: "Oplin will not store the SMB address that was assigned in the SMB ARP process in it EEPROM, so in the next power-up it will return to its EPROM default SMB address." ?4.2.1.7.2) • Changed Uncorrectable Error Severity bit 20 to be default 0 per PCIe* specification v1.1 ?4.1.9.8.1.4) • Changed and updated fields in MDFTC2 and added MDFTS2. (?13.2.38, ?13.2.41) • Added indication that AIT register is latched high cleared on read. (?9.2.3.12.16) • Added FW_reset_En bit to the Manageability Capability / Manageability Enable EERPOM register (?7.5.1.4) • Flash frequency was updated to 10Mhz according to the design implementation (?11.4.3.4) • Changed defaults of some PCS reserved bits • Added RxDPipeSize bits in RDRXCTL register • Updated MDIO buffers to be regular buffers to match the design and removed the need for external pull-ups for these pins • Added ANLP1.ANAS field • Added APBACE bit to GCR • Changed diagnostic registers for PB read/write data/descriptor pointers to match the current design • Included DCN024 Change EEPROM HW pointers to byte pointers • Included DCN026 Remove FML • Include DCN019 Remove VLAN classification support for VMDq • Include DCN025 Change maximum Rx buffer size support • Included DCN RMII multicast filtering support

Rev	Date		Comments
1.12			• Updated the EAS to properly reflect the fact that MDEF[6] is Multicast AND, as it was not consistent throughout the document • Minr typos and updates at different manageability sections • Updated EAS to reflect that only 4 SDP are loaded from EEPROM and corrected the SDP control registers in section 1.6.9 • Updated the device initialization sequence to reflect that PCIe* analog registers are only loaded after PE_RST_N • Updated EEC.FWE bits to be consistent (10 = enabled) • Added mapping from EEPROM fields to PCI configuration registers • Changed GCR (bit 20) Auto PBA Clear Enable to be Auto PBA Clear Disable (changed polarity) • Added explanation of how to use GSSR register • Added manageability host slave interface command description • Fixed a typo in the BSIZEHEADER definition to indicate it needs to be defined if DESCTYPE is greater or equal to 2 • Added Customer pin names to the pin interface tables. • Added description of the DFT registers in the diagnostic registers section • RMPN/TDHMPN ECC bits changed to be reserved. • FRTimer should not be used by Oplin SW. • Made the CB bits in GCR reserved (no need for CB unlock VDM) • Removed IXSM bit from Rx descriptors (not supported). • MDIO auto scan de-featured from Oplin.
1.13	12/06		• Fixed the typo in the default number of MSI-X vectors supported to 0x13. • Added a note to clarify that SW needs to update the tail descriptor on a packet boundary. • Updated the RSENSE pin description • Made the IPID_15 bit as reserved • Typo fixes. • Added reserved restriced not for external documentation reference to the de-featured capabilities
1.15	03/07		• Indicate that only 16 MSI-X vectors should be used (Oplin still supports 20 vectors but for future compatibility only 16 are exposed). • Replaced TPBTE to BPBFSM is the relevant place (enable bit name change) • Updated Tcase Max value to the right value (105C) • Reduced the max number of descriptors per ring to 32K-8. • Updated MDC speeds at 1Gbps • Removed (modified to reserved) ISCSI_DIS and ISCSI_DWC from RFCTL as they are not implemented in the design • Added optimal performance configurations • Added the Dummy Function Enable bit to NVM and proper description in chapter 5
1.2			• MNG chapter was updated by Assaf Agmon (major changes) • Power numbers updated
1.25			• Some fixes
	08/07		• Fixing "SERDES Reference clock specification 11.4.6"
1.25.1	08/07		• Patching "Power on sequence 11.3.1.1" chapter to limit 1.8V to 1.2V distance to 400uSec
1.25.2	08/07		• Patching "Power on sequence 11.3.1.1" chapter to increase 1.8V to 1.2V delay requirement to 500uSec
1.27	09/07		• Updated the Supported receive checksum capabilities table to indicate no TCP/UDP checksum offload for tunneled packets • Added the restriction to disable the music arbiter while changing music configuration • Added the restriction to do music configuration after device global reset. • Updated the music Tx arbiter scheme to match the HW implemnetation
1.35	01/08		• Updated to match approved DCN047 - changing the 64 bits statistics to 32 bits • Fixed a typo in the bit description of EEPROM control word 2 (section 7.3.1.2) • Added NVM word 0x2B - SW Phy Init Pointer • Added JTDO PU indication at active time

Rev	Date		Comments
1.36	05/08		Volume 1 Updates/Changes/Modifications • Updated Table 1-5 (starting with Rx Header Replication down to Relax Ordering • Updated RCLKEXTP/RCLKEXTN pin description (changed +/-0.0 15% to 0 ppm. • Updated section 3.1.6. Changed SMBCLK, SMBD, and SMBALRT_N types from T/s to o/d. • Added /FML to the note following section 3.1.6. • Updated the note following section 3.1.7. • Updated section 3.1.11. Changed customer pin name RSVDB17_NC to LAN_PWR_GOOD and customer pin name RSVDAC3_NC to POR_BYPASS. • Changed customer pin name RSVDNC to J1_VSS in section 3.1.11. • Changed Type column for JTDO from T/s to o/d. • Added "Internal" to Power Up and Active columns in section 3.2. • Added "external pull-up" to JTDO comment column in section 3.2. • Updated third bullet in section 4.1.3.2. • Updated Table 4-12. Replaced "v" with "b" for bits 5:3). • Updated Device Control2 Byte, Offset 0xA8, (RW) table on page 106. Included new text description for bit 15. • Updated section 4.4.4. Changed LAN "A" to LAN "0" and LAN "B" to LAN "1". • Updated "Software Reset", Link Reset" paragraphs on page 227/228. Removed "Bits that are normally . . .". • Updated section 5.3.3.5. Changed "PSRCTL" to "SRRCTL". • Deleted "and all descriptors are written back" from the second bullet under "Disabling" on page 239. Added third bullet "Wait until all descriptors . . . ". Volume 2 Updates/Changes/Modifications • Removed "See PKTTYPE table for . . . " from the Rsv, IPCS, L4CS, UDPCS table on page 29. • Removed "See PKTTYPE table for . . . " from the IPCS, L4CS, UDPCS table on page 36. • Changed PSRCTL to SRRCTL under the HBO section on page 37 (step 1). • Replaced "Large Send" to "TCP Segmentation Offload" under the IFCS section on page 62. • Replaced "Large Send" to "TCP Segmentation Offload" under the PAYLEN section on page 70. • Replaced "Large Send" to "TCP Segmentation Offload" in section 8.2.4. • Replaced "Large Send" to "TCP Segmentation Offload" on page 75. • Replaced "Large Send" to "TCP Segmentation Offload" in section 8.2.4.7. • Changed TCD fields from two to three in section 8.2.5.2. • Changed suggested range to 651 on page 101. • Changed LSO to TSO on page 131 (three places) and one place on page 134. • Added the MAC Manageability Control register to Table 9-2. • Deleted "An initial suggested range is 65- . . . " from page 222 (right before section 9.2.3.5.8. • Changed TDBHA to TDBAH on page 250. • Changed LSO to TSO on page 252. • Changed "configurations" to configuration in the SIZE description table in section 9.2.3.8.11. • Add 0x0 to the Initial Value field for bits 31:14 in section 9.2.3.10.2. • Fixed two typos in the LPHD and LPFD bit descriptions in section 9.2.3.12.7. • Removed "AP" from the bit 0 field in section 9.2.3.14.1. • Added an "*" to the VCC1P8 and VCC1P2 symbols. • Updated section 11.4.2.1. Added new total device power numbers and table notes a, b, and c. • Added signal JRST_N to the paragraph just before section 11.4.2.3, • Added zeros to SMBD and SMBCLK signals on page 366. Just before section 11.4.2.4. • Updated section 11.4.2.4. Added new RMII Input and Output Pads DC specifications. Also, deted the two notes just before section 11.4.3. • Updated section 11.4.3.4. Added new Min and Typ values. • Updated section 11.4.3.5. Added new Min value.

Rev	Date		Comments
1.37	06/08		• Deleted "Pu" from LAN1_DIS_N and LAN0_DIS_N Type column in section 3.1.11. • Deleted "Pu" from the JTDI, JTMS, and JRST_N Type column in section 3.1.12. • Updated Table 3-2. Added pull-up and pull-down information for RMII_CRS_NV, RMII_RXD[0], RMII_RXD[0], MDIO[0], MDIO[1], EE_CS_N, FLSH_CD_N, and JTCK. • Added "470 ohm" to JRST_N coment columns in section 3.2. • Updated "Revision" paragraph on page 89. Replaced word 0x1D to offset 11 - Device Revision ID. • Updated "Memory Address Space" description on page 90. Replaced word 0x0F to offset 7 - PCIe Control. • Updated section 4.1.9.3.3. Replaced words 0x24 and 0x14 with "at offset 7 - PCIe Control". • Updated "Address" description on page 93. Replaced word 0x0F with "set by the EEPROM at offset 7 (PCIe Control)". • Updated "Interrupt Pin" paragraph. Replaced words 0x24 and 0x14 with "at offset 1, EEPROM Configuration Space 0/1,". • Updated section 4.2.1.3. Replaced "(in the LRXEN1) word to "at offset 31 - LAN 0 Receive Enable 1". • Updated "Vendor Specific ID" paragraph on page 133. Replaced "words 7.3.5.2 in" to "at offset 1-3 (Ethernet Address)". • Updated "CBDM" description on page 137. Replaced "words 0x0-0x2" with "at offset". • Updated paragraph on page 143 beginning with "Oplin responds to one of the commands . . ". • Updated section 6.2.4.2. Replaced "7.3.1.2 word" to "at address 0x00 (EEPROM Control Word 1)". • Updated second paragraph in section 6.2.5.1 beginning with "Enable Port 1/0 . . . ". • Updated third and fourth bullet in section 6.2.5.4.2 beginning with "as loaded from the EEPROM . . . ". • Deleted the RXE bit description from page 37. • Updated the "RST" bit description in Device Control Register table (bit 26). • Updated the Unlock_EEP bit description in the 9.2.3.4.16Firmware Semaphore Register (bit 28). • Updated sections 9.2.3.13.49 and 9.2.3.13.509. Included new RQSMR and TQSMR descriptions. • Added "- (for both read and write-back formats)" to STA field description on page 69.
2.0	12/08		• Converted to single source using FrameMaker. • Updated internal/external pull-up/pull-down information.
2.2	February 2008		Initial Release (Intel Public).
2.3	May 2008		Updated sections/figures/tables: Product Features, 1.9.9, 2.2, 2.3, 2.9, 2.13, Table 2-17, 3.1.1.6, Table 3-17, 3.1.1.14.4, Note after Table 3-25, Device Cap, Device Control, Device Status, Link CAP Lin Control, Link Status, tables in section 3.1.1.14.6, 3.1.3.1, Figure 3-10, 3.2.1.2, 3.2.1.3, Table 3-38, Table 3-39, 3.2.3.2.5, Figure 3-12, Figure 3-13, 3.3.2.3.1.7, Table 3-45, 3.4.3.1.1, 3.4.3.1.2, 3.4.3.4.2, 3.4.3.6.2, Figure 3-20, Figure 3-21, Table 3-54, 3.5.2.4, 3.5.2.8, 3.5.2.10.1, 3.5.2.11, Table 3-59, Table 3-69, 3.5.3.3.2, 3.5.4.1, 3.5.4.2, Table 4-2, Table 4-4, 4.4.3.3.1 through 4.4.3.7, 4.4.3.3.12, 4.4.3.5.7 through 4.4.5.9, 4.4.3.6.2, 4.4.3.6.8, 4.4.3.6.11, 4.4.3.9, 4.4.3.11.1, 4.4.3.13.5, 4.4.3.13.8 through 4.4.3.13.10, 4.4.3.13.24, Table 5-7 through Table 5-10, Table 5-12, and 5.6.4, 5.6.6.
2.4	August 2008		Replaced Large Send Offload (LSO) with TCP Segmentation Offload (TSO). Removed all references to "Header Replication". Updated reference schematics. Updated Tables: 2-12, 2-14, 2-16, 3-17, 3-58, 3-71, 4-4, 5-3 through 5-5, 5-12, 5-15, 5-19, Updated Section: 1.2, 1.8, 1.9.10, 2.13, 3.1.1.10.1, 3.1.1.14.2, 3.2.2.14.3.1, 3.1.1.14.4, 3.1.4.3.4, 3.2.1.3, 3.3.1.3.2, 3.3.1.4.1, 3.3.1.4.4.2, 3.4 through 3.4.4, 3.5.2, 4.4.3.1.1, 4.4.3.3.7, 4.4.3.5.7, 4.4.3.6.4, 4.4.3.9.49, 4.4.3.9.50, 5.4.3, 5.5.1, 7.6, 7.8, 7.13.2.



Rev	Date		Comments
2.5	November 2008		Updated Section: 3.2.1.2, 3.4.2.2.8, 3.4.2.3.2, 3.4.2.3.3, 3.4.2.5.2, 3.4.2.5.4, 3.4.3, 3.5.2.13, 3.5.3.2, 3.5.3.3.1.1, 3.5.3.3.1.3, 3.5.5.2, 3.5.5.3.1. 3.5.5.3.2, 3.5.6.1, 3.5.7, 4.4.3.1.6, 4.4.3.5.7, 4.4.3.6.13, 4.4.3.7.6, 4.4.3.9.5, 4.4.3.10.5, 4.4.3.10.6, 4.4.3.10.7, 4.4.3.10.8, 4.4.3.10.9, 4.4.3.13.8, 4.4.3.13.15, 4.4.3.13.23, 4.4.3.13.24, 4.4.3.13.25, 5.4.2, 5.5.1, 5.5.2, 5.6.6, and 5.6.6.1. Updated Table 5-2, 5-6 and supported figure. Updated Figure 5-1 notes. Added Section 7.16.8.
3.1	April 2009		Section 4.4.3.5.12, Drop Enable Control – DROPEN (0x03D04 – 0x03D08; RW) - Description updated for clarity. Section 5.3.12.9, Still Having Problems? & Section 5.3.12.5.1, Firmware Semaphore Register (FWSM, 0x10148) - FWSM indicated in both places. 0x10148 is the correction. Section 5.3.13, Sample Configurations - Sample filtering configurations added. Section 9.1.1, GHOST ECC Register - GHECCR (0x110B0, RW). Diagnostic register added to public documentation because it has limited public use as a workaround. Support for PCIe* Statistics Counters dropped.
3.2	October 2010		Section 3.4.2.2, PBA Number Module – Words 0x15:0x16. Updated to reflect new methodology.

Contents

1. General Information 31

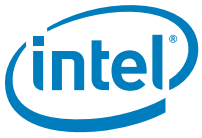
1.1 Introduction 31
1.2 Terminology and Acronyms 31
1.3 Reference Documents 33
1.4 Models and Symbols 35
1.5 Physical Layer Conformance Testing 35
1.6 Design and Board Layout Checklists 35
1.7 Number Conventions 35
1.8 System Configurations 35
1.9 External Interfaces 36
1.9.1 PCIe Interface 37
1.9.2 XAUI Interfaces 37
1.9.3 EEPROM Interface 38
1.9.4 Serial Flash Interface 39
1.9.5 SMBus Interface 39
1.9.6 NC-SI Interface 39
1.9.7 MDIO Interfaces 39
1.9.8 Software-Definable Pins (SDP) Interface (General-Purpose I/O) 40
1.9.9 LED Interface 40

2. Signal Descriptions and Pinout List 41

2.1 Signal Type Definitions 41
2.2 PCIe Interface 42
2.3 XAUI Interface Signals 43
2.4 EEPROM and Serial Flash Interface Signals 45
2.5 SMBus and NC-SI Signals 45
2.6 MDI/O Signals 46
2.7 Software-Definable Pins 46
2.8 LED Signals 47
2.9 Miscellaneous Signals 47
2.10 Test Interface Signals 48
2.11 Power Supplies 48
2.12 Alphabetical Pinout/Signal Name 50
2.13 Internal/External Pull-Up/Pull-Down Specifications 61
2.14 Pin Assignments (Ball Out) 63

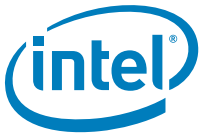
3. Functional Description 69

3.1 Interconnects 69
3.1.1 PCIe 69
3.1.1.1 Architecture, Transaction and Link Layer Properties 70
3.1.1.1.1 Physical Interface Properties 71
3.1.1.1.2 Advanced Extensions 71
3.1.1.2 General Functionality 71
3.1.1.2.1 Native/Legacy 71
3.1.1.2.2 Locked Transactions 71
3.1.1.2.3 End-to-End CRC (ECRC) 71
3.1.1.3 Host Interface 71
3.1.1.3.1 Tag ID Allocation 72
3.1.1.3.2 Completion Timeout Mechanism 74
3.1.1.3.2.1 Completion Timeout Enable 74
3.1.1.3.2.2 Resend Request Enable 74
3.1.1.3.2.3 Completion Timeout Period 75
3.1.1.4 Transaction Layer 75
3.1.1.4.1 Transaction Types Accepted 76
3.1.1.4.1.1 Partial Memory Read and Write Requests 76
3.1.1.4.2 Transaction Types Initiated 77



3.1.1.4.2.1 Data Alignment 77
3.1.1.4.2.2 Multiple Tx Data Read Requests (MULR) 78
3.1.1.5 Messages 78
3.1.1.5.1 Received Messages 78
3.1.1.5.2 Transmitted Messages 78
3.1.1.6 Ordering Rules 79
3.1.1.6.1 Out of Order Completion Handling 80
3.1.1.7 Transaction Definition and Attributes 80
3.1.1.7.1 Max Payload Size 80
3.1.1.7.2 Traffic Class (TC) and Virtual Channels (VC) 80
3.1.1.7.3 Relaxed Ordering 80
3.1.1.7.4 No Snoop 80
3.1.1.7.5 No Snoop and Relaxed Ordering for LAN Traffic 81
3.1.1.7.5.1 No Snoop Option for Payload 81
3.1.1.8 Flow Control 81
3.1.1.8.1 82598 Flow Control Rules 81
3.1.1.8.2 Upstream Flow Control Tracking 82
3.1.1.8.3 Flow Control Update Frequency 82
3.1.1.8.4 Flow Control Timeout Mechanism 82
3.1.1.9 Error Forwarding 83
3.1.1.10 Link Layer 83
3.1.1.10.1 ACK/NAK Scheme 83
3.1.1.10.2 Supported DLLPs 83
3.1.1.10.3 Transmit EDB Nullifying 84
3.1.1.11 PHY 84
3.1.1.11.1 Link Speed 84
3.1.1.11.2 Link Width 84
3.1.1.11.3 Polarity Inversion 84
3.1.1.11.4 L0s Exit Latency 85
3.1.1.11.5 Lane-to-Lane De-Skew 85
3.1.1.11.6 Lane Reversal 85
3.1.1.11.7 Reset 87
3.1.1.11.8 Scrambler Disable 87
3.1.1.12 Error Events and Error Reporting 87
3.1.1.12.1 General Description 87
3.1.1.12.2 Error Events 88
3.1.1.12.3 Error Pollution 90
3.1.1.12.4 Completion With Unsuccessful Completion Status 90
3.1.1.12.5 Error Reporting Changes 90
3.1.1.13 Performance Monitoring 91
3.1.1.14 Configuration Registers 91
3.1.1.14.1 PCI Compatibility 91
3.1.1.14.2 Configuration Sharing Among PCI Functions 92
3.1.1.14.3 Mandatory PCI Configuration Registers 93
3.1.1.14.3.1 Expansion ROM Base Address 98
3.1.1.14.4 PCI Power Management Registers 99
3.1.1.14.5 MSI Configuration 102
3.1.1.14.6 MSI-X Configuration 103
3.1.1.14.7 PCIe Configuration Registers 107
3.1.1.14.8 PCIe Extended Configuration Space 115
3.1.1.14.8.1 Advanced Error Reporting Capability 116
3.1.1.14.8.2 Serial Number 120
3.1.2 Manageability Interfaces (SMBus/NC-SI) 122
3.1.2.1 SMBus Pass-Through Interface 123
3.1.2.1.1 General 123
3.1.2.1.2 Pass-Through Capabilities 123
3.1.2.2 NC-SI 123
3.1.2.2.1 Interface Specification 123
3.1.2.2.2 Electrical Characteristics 124
3.1.2.2.3 NC-SI Transactions 124

3.1.2.2.3.1 NC-SI-SMBus Mode 124
3.1.2.2.3.2 NC-SI Mode 124
3.1.3 Non-Volatile Memory (EEPROM/Flash) 125
3.1.3.1 EEPROM 125
3.1.3.1.1 Software Accesses 125
3.1.3.1.2 Signature Field 126
3.1.3.1.3 Protected EEPROM Space 126
3.1.3.1.3.1 Initial EEPROM Programming 126
3.1.3.1.3.2 EEPROM Protected Areas 126
3.1.3.1.3.3 Activating the Protection Mechanism 126
3.1.3.1.3.4 Non Permitted Accesses to Protected Areas in the EEPROM 127
3.1.3.1.4 EEPROM Recovery 127
3.1.3.2 Flash 128
3.1.3.2.1 Flash Interface Operation 128
3.1.3.2.2 Flash Write Control 129
3.1.3.2.3 Flash Erase Control 129
3.1.3.2.4 Flash Access Contention 129
3.1.4 Network Interface 129
3.1.4.1 10 GbE Interface 129
3.1.4.1.1 XGXS – PCS/PMA 130
3.1.4.2 GbE Interface 131
3.1.4.3 Auto Negotiation and Link Setup Features 131
3.1.4.3.1 Link Configuration 131
3.1.4.3.2 MAC Link Setup and Auto Negotiation 132
3.1.4.3.3 Hardware Detection of Non-Auto Negotiation Partner 132
3.1.4.4 MDIO/MDC 132
3.1.4.4.1 MDIO Direct Access 132
3.1.4.5 Ethernet (Legacy) Flow Control 133
3.1.4.5.1 MAC Control Frames and Reception of Flow Control Packets 133
3.1.4.5.2 Discard Pause Frames and Pass MAC Control Frames 134
3.1.4.5.3 Transmission of Pause Frames 134
3.1.4.6 MAC Speed Change at Different Power Modes 135
3.2 Initialization 136
3.2.1 Power Up 136
3.2.1.1 Power-Up Sequence 136
3.2.1.2 Power-Up Timing Diagram 138
3.2.1.2.1 Timing Requirements 139
3.2.1.2.2 Timing Guarantees 139
3.2.1.3 Reset Operation 140
3.2.2 Specific Function Enable/Disable 143
3.2.2.1 General 143
3.2.2.2 Overview 143
3.2.2.3 Event Flow for Enable/Disable Functions 144
3.2.2.3.1 BIOS Disable the LAN Function at Boot Time by Using Strapping Option 144
3.2.2.3.2 Multi-Function Advertisement 145
3.2.2.3.3 Interrupt Use 145
3.2.2.3.4 Power Reporting 145
3.2.2.4 Device Disable Overview 145
3.2.2.4.1 BIOS Disable the Device at Boot Time by Using Strapping Option 145
3.2.3 Software Initialization and Diagnostics 146
3.2.3.1 Power Up State 146
3.2.3.2 Initialization Sequence 146
3.2.3.2.1 Disabling Interrupts During Initialization 146
3.2.3.2.2 Global Reset and General Configuration 146
3.2.3.2.3 Link Setup Mechanisms and Control/Status Bit Summary 147
3.2.3.2.3.1 BX 1 Gb/s Link Setup 147
3.2.3.2.3.2 10 Gb/s Link Setup 147
3.2.3.2.4 Initialization of Statistics 148
3.2.3.2.5 Receive Initialization 148
3.2.3.2.6 Dynamic Enabling and Disabling of Receive Queues 149

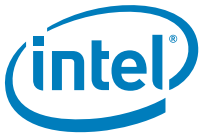


3.2.3.2.7 Transmit Initialization 149
3.2.3.2.8 Dynamic Enabling and Disabling of Transmit Queues 150
3.3 Power Management and Delivery 150
3.3.1 Power Delivery 150
3.3.1.1 82598 Power States 150
3.3.1.2 Auxiliary Power Usage 151
3.3.1.3 Interconnects Power Management 152
3.3.1.3.1 PCIe Link Power Management 152
3.3.1.3.2 Network Interfaces Power Management 154
3.3.1.4 Power States 154
3.3.1.4.1 D0 Uninitialized State 154
3.3.1.4.1.1 Entry into D0u State 154
3.3.1.4.2 D0active State 154
3.3.1.4.2.1 Entry to D0a State 155
3.3.1.4.3 D3 State (PCI-PM D3hot) 155
3.3.1.4.3.1 Entry to D3 State 155
3.3.1.4.3.2 Master Disable 155
3.3.1.4.4 Dr State 156
3.3.1.4.4.1 Dr Disable Mode 156
3.3.1.4.4.2 Entry to Dr State 157
3.3.1.5 Timing of Power-State Transitions 157
3.3.1.5.1 Transition from D0a to D3 and back without PE_RST_N 158
3.3.1.5.2 Transition from D0a to D3 and Back with PE_RST_N 159
3.3.1.5.3 Transition from D0a to Dr and Back Without Transition to D3 161
3.3.1.5.4 Timing Requirements 161
3.3.1.5.5 Timing Guarantees 162
3.3.2 Wake Up 163
3.3.2.1 Advanced Power Management Wake Up 163
3.3.2.2 ACPI Power Management Wakeup 164
3.3.2.3 Wake-Up Packets 164
3.3.2.3.1 Pre-Defined Filters 165
3.3.2.3.1.1 Directed Exact Packet 165
3.3.2.3.1.2 Directed Multicast Packet 165
3.3.2.3.1.3 Broadcast 165
3.3.2.3.1.4 Magic Packet* 166
3.3.2.3.1.5 ARP/IPv4 Request Packet 167
3.3.2.3.1.6 Directed IPv4 Packet 168
3.3.2.3.1.7 Directed IPv6 Packet 168
3.3.2.3.2 Flexible Filter 169
3.3.2.3.2.1 IPX Diagnostic Responder Request Packet 169
3.3.2.3.2.2 Directed IPX Packet 170
3.3.2.3.2.3 IPv6 Neighbor Discovery Filter 170
3.3.2.3.3 Wake-Up Packet Storage 170
3.4 NVM Map (EEPROM) 171
3.4.1 EEPROM General Map 171
3.4.2 EEPROM Software Section 172
3.4.2.1 Compatibility Fields – Words 0x10-0x14 172
3.4.2.2 PBA Number Module – Words 0x15:0x16 172
3.4.2.3 Software EEGEN Work Area 174
3.4.2.3.1 DS_Version – Word 0x29 174
3.4.2.3.2 OEM Version and ID – Word 0x2A 174
3.4.2.3.3 Software Init Section Pointer – Word 0x2 174
3.4.2.3.4 eTrack_ID – Word 0x2D:2E 174
3.4.2.4 PXE Configuration Words – Word 0x30:3B 175
3.4.2.4.1 Setup Options PCI Function 0 – Word 0x30 175
3.4.2.4.2 Configuration Customization Options PCI Function 0 – Word 0x31 176
3.4.2.4.3 PXE Version – Word 0x32 177
3.4.2.4.4 IBA Capabilities – Word 0x33 177
3.4.2.4.5 Setup Options PCI Function 1 – Word 0x34 177
3.4.2.4.6 Configuration Customization Options PCI Function 1 – Word 0x35 177

3.4.2.4.7 Setup Options PCI Function 2 – Word 0x38 177
3.4.2.4.8 Configuration Customization Options PCI Function 2 – Word 0x39 178
3.4.2.4.9 Setup Options PCI Function 3 – Word 0x3A 178
3.4.2.4.10 Configuration Customization Options PCI Function 3 – Word 0x3B 178
3.4.2.5 EEPROM Checksum Calculation 178
3.4.3 Hardware EEPROM Sections 179
3.4.3.1 EEPROM Init Section 179
3.4.3.1.1 EEPROM Control Word 1 – Address 0x00 179
3.4.3.1.2 EEPROM Control Word 2 – Address 0x01 180
3.4.3.1.3 EEPROM Control Word 3 – Address 0x38 180
3.4.3.2 EEPROM Hardware Pointers 180
3.4.3.2.1 Analog Configuration Sections – Words 0x04:0x05 180
3.4.3.2.1.1 EEPROM Analog Configuration – Section Length 181
3.4.3.2.1.2 EEPROM Analog Configuration – Data Word 181
3.4.3.2.2 PCIe Analog Pointer – Word 0x03 181
3.4.3.2.2.1 PCIe Analog – Section Length 181
3.4.3.2.2.2 PCIe Analog Selector 181
3.4.3.2.2.3 PCIe Analog Word 182
3.4.3.3 EEPROM PCIe General Configuration Section 182
3.4.3.3.1 PCIe General Configuration – Section Length 183
3.4.3.3.2 PCIe Init Configuration 1 – Offset 1 183
3.4.3.3.3 PCIe Init Configuration 2 – Offset 2 185
3.4.3.3.4 PCIe Init Configuration 3 – Offset 3 185
3.4.3.3.5 PCIe Control – Offset 4 186
3.4.3.3.6 PCIe Control – Offset 5 187
3.4.3.3.7 PCIe Control – Offset 6 – LAN Power Consumption 188
3.4.3.3.8 PCIe Control – Offset 7 188
3.4.3.3.9 PCIe Control – Offset 8 – Sub-System ID 188
3.4.3.3.10 PCIe Control – Offset 9 – Sub-System Vendor ID 189
3.4.3.3.11 PCIe Control – Offset 10 – Dummy Device ID 189
3.4.3.3.12 PCIe Control – Offset 11 – Device Revision ID 189
3.4.3.4 EEPROM PCIe Configuration Space 0/1 Sections 189
3.4.3.4.1 PCIe Configuration Space 0/1 – Section Length 189
3.4.3.4.2 EEPROM PCIe Configuration Space 0/1- Offset 1 190
3.4.3.4.3 EEPROM PCIe Configuration Space 0/1 – Offset 2 Device ID 190
3.4.3.5 EEPROM Core 0/1 Section 190
3.4.3.5.1 Core Configuration Section – Section Length 191
3.4.3.5.2 Ethernet Address – Offset 1-3 191
3.4.3.5.3 LEDs Configuration – Offset 4-5 192
3.4.3.5.4 SDP Control – Offset 6 192
3.4.3.5.5 Filter Control – Offset 7 193
3.4.3.6 EEPROM MAC 0/1 Sections 193
3.4.3.6.1 MAC Configuration Section – Section Length 193
3.4.3.6.2 Link Mode Configuration – Offset 1 194
3.4.3.6.3 SWAP Configuration – Offset 2 195
3.4.3.6.4 Swizzle and Polarity Configuration – Offset 3 196
3.4.3.6.5 Auto Negotiation Defaults – Offset 4 197
3.4.3.6.6 AUTOC2 – Upper Half– Offset 5 198
3.4.4 Hardware Section – Auto-Read 198
3.4.5 Manageability Control Sections 199
3.4.5.1 Common Firmware Pointers 199
3.4.5.1.1 Test Configuration Pointer – (Global Offset 0x0) 200
3.4.5.1.2 Loader Patch Pointer – (Global Offset 0x1) 200
3.4.5.1.3 No Manageability Patch Pointer – (Global Offset 0x2) 200
3.4.5.1.3.1 Manageability Capability/Manageability Enable – (Global Offset 0x3) 200
3.4.5.1.3.2 NC-SI Configuration Pointer – (Global Offset 0x4) 201
3.4.5.1.4 Test Structure 201
3.4.5.1.4.1 Section Header – (Offset 0x0) 201
3.4.5.1.4.2 Loopback Test Configuration – (Offset 0x1) 201
3.4.5.1.5 Patch Structure 201

3.4.5.1.5.1 Patch Data Size – (Offset 0x0) 201
3.4.5.1.5.2 Block CRC8– (Offset 0x1) 202
3.4.5.1.5.3 Patch Entry Point Pointer Low Word – (Offset 0x2) 202
3.4.5.1.5.4 Patch Entry Point Pointer High Word – (Offset 0x3) 202
3.4.5.1.5.5 Patch Version 1 Word – (Offset 0x4) 202
3.4.5.1.5.6 Patch Version 2 Word – (Offset 0x5) 202
3.4.5.1.5.7 Patch Version 3 Word – (Offset 0x6) 203
3.4.5.1.5.8 Patch Version 4 Word – (Offset 0x7) 203
3.4.5.1.5.9 Patch Data Words – (Offset 0x8 – Block Length) 203
3.4.5.1.6 Pass Through Control Words 203
3.4.5.1.6.1 Pass Through Pointers 203
3.4.5.1.6.1.1 Pass Through Patch Configuration Pointer – (Global Offset 0x4) 203
3.4.5.1.6.1.2 Pass Through LAN 0 Configuration Pointer – (Global Offset 0x5) 203
3.4.5.1.6.1.3 Sideband Configuration Pointer – (Global Offset 0x6) 204
3.4.5.1.6.1.4 Flexible TCO Filter Configuration Pointer – (Global Offset 0x7) 204
3.4.5.1.6.1.5 Pass Through LAN 1 Configuration Pointer – (Global Offset 0x8) 204
3.4.5.1.6.1.6 NC-SI Microcode Download Pointer – (Global Offset 0x9) 204
3.4.5.1.6.2 Pass Through LAN Configuration Structure 204
3.4.5.1.6.2.1 Section Header – (0ffset 0x0) 204
3.4.5.1.6.2.2 LAN 0 IPv4 Address 0 (LSB) MIPAF0 – (0ffset 0x01) 204
3.4.5.1.6.2.3 LAN 0 IPv4 Address 0 (MSB) (MIPAF0) – (0ffset 0x02) 205
3.4.5.1.6.2.4 LAN 0 IPv4 Address 1 MIPAF1 – (0ffset 0x03-x004) 205
3.4.5.1.6.2.5 LAN 0 IPv4 Address 2 MIPAF2 – (0ffset 0x05-0x06) 205
3.4.5.1.6.2.6 LAN 0 IPv4 Address 3 MIPAF3 – (0ffset 0x07-0x08) 205
3.4.5.1.6.2.7 LAN 0 MAC Address 0 (LSB) MMAL0 – (0ffset 0x09) 205
3.4.5.1.6.2.8 LAN 0 MAC Address 0 (LSB) MMAL0 – (0ffset 0x0A) 205
3.4.5.1.6.2.9 LAN 0 MAC Address 0 (MSB) MMAH0 – (0ffset 0x0B) 205
3.4.5.1.6.2.10 LAN 0 MAC Address 1 MMAL/H1 – (0ffset 0x0C-0x0E) 205
3.4.5.1.6.2.11 LAN 0 MAC Address 2 MMAL/H2 – (0ffset 0x0F-0x11) 206
3.4.5.1.6.2.12 LAN 0 MAC Address 3 MMAL/H3 – (0ffset 0x12-0x14) 206
3.4.5.1.6.2.13 LAN 0 UDP Flexible Filter Ports 0 – 15 (MFUTP Registers) – (0ffset 0x15-0x24) 206
3.4.5.1.6.2.14 LAN 0 VLAN Filter 0 – 7 (MAVTV Registers) – (0ffset 0x25 – 0x2C) 206
3.4.5.1.6.2.15 LAN 0 Manageability Filters Valid (MFVAL LSB) – (0ffset 0x2D) 206
3.4.5.1.6.2.16 LAN 0 Manageability Filters Valid (MFVAL MSB) – (0ffset 0x2E) 206
3.4.5.1.6.2.17 LAN 0 MANC Value LSB (LMANC LSB) – (0ffset 0x2F) 207
3.4.5.1.6.2.18 LAN 0 MANC Value MSB (LMANC MSB) – (0ffset 0x30) 207
3.4.5.1.6.2.19 LAN 0 Receive Enable 1 (LRXEN1) – (0ffset 0x31) 207
3.4.5.1.6.2.20 LAN 0 Receive Enable 2 (LRXEN2) – (0ffset 0x32) 208
3.4.5.1.6.2.21 LAN 0 MANC2H Value (LMANC2H LSB) – (0ffset 0x33) 208
3.4.5.1.6.2.22 LAN 0 MANC2H Value (LMANC2H MSB) – (0ffset 0x34) 208
3.4.5.1.6.2.23 Manageability Decision Filters- MDEF0 (1) – (0ffset 0x35) 208
3.4.5.1.6.2.24 Manageability Decision Filters- MDEF0 (2) – (0ffset 0x36) 209
3.4.5.1.6.2.25 Manageability Decision Filters- MDEF1-6 (1-2) – (0ffset 0x37-0x42) 209
3.4.5.1.6.2.26 ARP Response IPv4 Address 0 (LSB) – (0ffset 0x43) 209
3.4.5.1.6.2.27 ARP Response IPv4 Address 0 (MSB) – (0ffset 0x44) 209
3.4.5.1.6.2.28 LAN 0 IPv6 Address 0 (LSB) (MIPAF) – (0ffset 0x45) 210
3.4.5.1.6.2.29 LAN 0 IPv6 Address 0 (MSB) (MIPAF) – (0ffset 0x46) 210
3.4.5.1.6.2.30 LAN 0 IPv6 Address 0 (LSB) (MIPAF) – (0ffset 0x47) 210
3.4.5.1.6.2.31 LAN 0 IPv6 Address 0 (MSB) (MIPAF) – (0ffset 0x48) 210
3.4.5.1.6.2.32 LAN 0 IPv6 Address 0 (LSB) (MIPAF) – (0ffset 0x49) 210
3.4.5.1.6.2.33 LAN 0 IPv6 Address 0 (MSB) (MIPAF) – (0ffset 0x4A) 211
3.4.5.1.6.2.34 LAN 0 IPv6 Address 0 (LSB) (MIPAF) – (0ffset 0x4B) 211
3.4.5.1.6.2.35 LAN 0 IPv6 Address 0 (MSB) (MIPAF) – (0ffset 0x4C) 211
3.4.5.1.6.2.36 LAN 0 IPv6 Address 1 (MIPAF) – (0ffset 0x4D-0x54) 211
3.4.5.1.6.2.37 LAN 0 IPv6 Address 2 (MIPAF) – (0ffset 0x55-0x5C) 211
3.4.5.1.7 SMBus Configuration Structure 211
3.4.5.1.7.1 Section Header – (0ffset 0x0) 211
3.4.5.1.7.2 SMBus Maximum Fragment Size – (0ffset 0x01) 212
3.4.5.1.7.3 SMBus Notification Timeout and Flags – (0ffset 0x02) 212

3.4.5.1.7.4 SMBus Slave Addresses – (0ffset 0x03) 212
3.4.5.1.7.5 Fail-Over Register (Low Word) – (0ffset 0x04) 213
3.4.5.1.7.6 Fail-Over Register (High Word) – (0ffset 0x05) 213
3.4.5.1.7.7 NC-SI Configuration (0ffset 0x06) 213
3.4.5.1.8 Flexible TCO Filter Configuration Structure 214
3.4.5.1.8.1 Section Header – (0ffset 0x0) 214
3.4.5.1.8.2 Flexible Filter Length and Control – (0ffset 0x01) 214
3.4.5.1.8.3 Flexible Filter Enable Mask – (0ffset 0x02 – 0x09) 214
3.4.5.1.8.4 Flexible Filter Data – (0ffset 0x0A – Block Length) 214
3.4.5.1.9 NC-SI Microcode Download Structure 215
3.4.5.1.9.1 Patch Data Size – (Offset 0x0) 215
3.4.5.1.9.2 Rx and Tx Code Size – (Offset 0x1) 215
3.4.5.1.9.3 Download Data – (Offset 0x2 – Data Size) 215
3.4.5.1.10 NC-SI Configuration Structure 215
3.4.5.1.10.1 Section Header (0ffset 0x0) 215
3.4.5.1.10.2 Rx Mode Control1 (RR_CTRL[15:0]) (Offset 0x1) 216
3.4.5.1.10.3 Rx Mode Control2 (RR_CTRL[31:16]) (Offset 0x2) 216
3.4.5.1.10.4 Tx Mode Control1 (RT_CTRL[15:0]) (Offset 0x3) 216
3.4.5.1.10.5 Tx Mode Control2 (RT_CTRL[31:16]) (Offset 0x4) 216
3.4.5.1.10.6 MAC Tx Control Reg1 (TxCntrlReg1 (15:0]) (Offset 0x5) 217
3.4.5.1.10.7 NC-SI Settings (NCSISET) (Offset 0x7) 217
3.5 Rx/Tx Functions 217
3.5.1 Device Data/Control Flows 217
3.5.1.1 Transmit Data Flow 217
3.5.1.2 Rx Data Flow 219
3.5.2 Receive Functionality 220
3.5.2.1 Packet Filtering 223
3.5.2.1.1 L2 Filtering 224
3.5.2.1.1.1 Unicast Filter 226
3.5.2.1.1.2 Multicast Filter (Partial) 226
3.5.2.1.2 VLAN Filtering 226
3.5.2.2 Intel® 82598 10 GbE Controller System Manageability Interface application noteReceive Data Storage 227
3.5.2.3 Legacy Receive Descriptor Format 228
3.5.2.4 Advanced Receive Descriptors 230
3.5.2.5 Receive UDP Fragmentation Checksum 237
3.5.2.6 Receive Descriptor Fetching 237
3.5.2.7 Receive Descriptor Write-Back 238
3.5.2.8 Receive Descriptor Queue Structure 238
3.5.2.9 Header Splitting and Replication 240
3.5.2.9.1 Purpose 240
3.5.2.9.2 Description 240
3.5.2.10 Receive-Side Scaling (RSS) 242
3.5.2.10.1 RSS Hash Function 244
3.5.2.10.1.1 Hash for IPv4 with TCP 246
3.5.2.10.1.2 Hash for IPv4 with UDP 246
3.5.2.10.1.3 Hash for IPv4 without TCP 246
3.5.2.10.1.4 Hash for IPv6 with TCP 246
3.5.2.10.1.5 Hash for IPv6 with UDP 247
3.5.2.10.1.6 Hash for IPv6 without TCP 247
3.5.2.10.2 Indirection Table 247
3.5.2.10.3 RSS Verification Suite 247
3.5.2.11 Receive Queuing for Virtual Machine Devices (VMDq) 248
3.5.2.11.1 Association Through MAC Address 248
3.5.2.12 Receive Checksum Offloading 248
3.5.3 Transmit Functionality 252
3.5.3.1 Packet Transmission 252
3.5.3.1.1 Transmit Data Storage 252
3.5.3.2 Transmit Contexts 252
3.5.3.3 Transmit Descriptors 253



3.5.3.3.1 Description 253
3.5.3.3.1.1 Legacy Transmit Descriptor Format 253
3.5.3.3.1.2 Advanced Transmit Context Descriptor 257
3.5.3.3.1.3 Advanced Transmit Data Descriptor 259
3.5.3.3.2 Transmit Descriptor Structure 261
3.5.3.3.3 Transmit Descriptor Fetching 263
3.5.3.3.4 Transmit Descriptor Write-Back 264
3.5.3.4 TCP Segmentation 264
3.5.3.4.1 Assumptions 264
3.5.3.4.2 Transmission Process 264
3.5.3.4.2.1 TCP Segmentation Performance 265
3.5.3.4.3 Packet Format 265
3.5.3.4.4 TCP Segmentation Indication 266
3.5.3.4.5 IP and TCP/UDP Headers 267
3.5.3.4.6 Transmit Checksum Offloading with TCP Segmentation 272
3.5.3.4.7 IP/TCP/UDP Header Updating 274
3.5.3.4.7.1 TCP/IP/UDP Header for the first Frames 274
3.5.3.4.7.2 TCP/IP/UDP Header for the Subsequent Frames 274
3.5.3.4.7.3 TCP/IP/UDP Header for the Last Frame 275
3.5.3.4.8 IP/TCP/UDP Checksum Offloading 276
3.5.3.5 IP/TCP/UDP Transmit Checksum Offloading in Non-Segmentation Mode 277
3.5.3.5.1 IP Checksum 277
3.5.3.5.2 TCP Checksum 277
3.5.3.6 Multiple Transmit Queues 278
3.5.3.6.1 Description 278
3.5.3.7 Transmit Completions Head Write Back 279
3.5.3.7.1 Description 279
3.5.4 Interrupts 279
3.5.4.1 Registers 279
3.5.4.2 Interrupt Moderation 281
3.5.4.3 Clearing Interrupt Causes 283
3.5.4.4 Dynamic Interrupt Moderation 283
3.5.4.4.1 Implementation 284
3.5.4.5 TCP Timer Interrupt 284
3.5.4.5.1 Description 284
3.5.4.6 MSI-X Interrupts 284
3.5.5 802.1q VLAN Support 285
3.5.5.1 802.1q VLAN Packet Format 285
3.5.5.2 802.1q Tagged Frames 286
3.5.5.3 Transmitting and Receiving 802.1q Packets 286
3.5.5.3.1 Adding 802.1q Tags on Transmits 286
3.5.5.3.2 Stripping 802.1q Tags on Receives 287
3.5.5.4 802.1q VLAN Packet Filtering 287
3.5.6 DCA 288
3.5.6.1 Description 288
3.5.6.2 PCIe Message Format for DCA (MWr Mode) 290
3.5.7 LED's 291

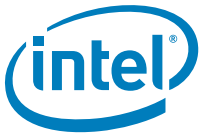
4. Programming Interface 293

4.1 Address Regions 293
4.2 Memory-Mapped Access 293
4.2.1 Memory-Mapped Access to Internal Registers and Memories 293
4.2.2 Memory-Mapped Accesses to Flash 293
4.2.3 Memory-Mapped Access to Expansion ROM 294
4.3 I/O-Mapped Access 294
4.3.1 IOADDR (I/O Offset 0x00, RW) 294
4.3.2 IODATA (I/O Offset 0x04, RW) 294
4.3.3 Undefined I/O Offsets 295
4.4 Device Registers 296
4.4.1 Terminology 296

4.4.2 Register List 296
4.4.3 Register Descriptions 304
4.4.3.1 General Control Registers 304
4.4.3.1.1 Device Control Register – CTRL (0x00000/0x00004, RW) 304
4.4.3.1.2 Device Status Register – STATUS (0x00008; R) 305
4.4.3.1.3 Extended Device Control Register – CTRL_EXT (0x00018; RW) 305
4.4.3.1.4 Extended SDP Control – ESDP (0x00020, RW) 306
4.4.3.1.5 Extended OD SDP Control – EODSDP (0x00028; RW) 307
4.4.3.1.6 LED Control – LEDCTL (0x00200; RW) 308
4.4.3.1.7 TCP_Timer - TCPTIMER (0x0004C, RW) 310
4.4.3.2 EEPROM/Flash Registers 310
4.4.3.2.1 EEPROM/Flash Control Register – EEC (0x10010; RW) 310
4.4.3.2.2 EEPROM Read Register – EERD (0x10014; RW) 313
4.4.3.2.3 Flash Access Register – FLA (0x1001C; RW) 313
4.4.3.2.4 Manageability EEPROM Control Register – EEMNGCTL (0x10110; RW) 314
4.4.3.2.5 Manageability EEPROM Read/Write Data – EEMNGDATA (0x10114; RW) 315
4.4.3.2.6 Manageability Flash Control Register – FLMNGCTL (0x10118; RW) 315
4.4.3.2.7 Manageability Flash Read Data – FLMNGDATA (0x1011C; RW) 316
4.4.3.2.8 Manageability Flash Read Counter – FLMNGCNT (0x10120; RW) 316
4.4.3.2.9 Flash Opcode Register – FLOP (0x01013C; RW) 316
4.4.3.2.10 General Receive Control – GRC (0x10200; RW) 317
4.4.3.3 Interrupt Registers 317
4.4.3.3.1 Extended Interrupt Cause Register EICR (0x00800, RC) 317
4.4.3.3.2 Extended Interrupt Cause Set Register EICS (0x00808, WO) 319
4.4.3.3.3 Extended Interrupt Mask Set/Read Register EIMS (0x00880, RWS) 319
4.4.3.3.4 Extended Interrupt Mask Clear Register EIMC (0x00888, WO) 320
4.4.3.3.5 Extended Interrupt Auto Clear Register EIAC (0x00810, RW) 321
4.4.3.3.6 Extended Interrupt Auto Mask Enable Register – EIAM (0x00890, RW) 322
4.4.3.3.7 Extended Interrupt Throttle Registers – EITR (0x00820 – 0x0086C, RW) 322
4.4.3.3.8 Interrupt Vector Allocation Registers IVAR (0x00900 + 4*n [n=0…24], RW) 323
4.4.3.3.9 MSI-X Table - MSIXT (BAR3: 0x00000 – 0x0013C, RW) 324
4.4.3.3.10 MSI-X Pending Bit Array – MSIXPBA (BAR3: 0x02000, RO) 324
4.4.3.3.11 MSI-X Pending Bit Array Clear – PBACL (0x11068, RW) 324
4.4.3.3.12 General Purpose Interrupt Enable – GPIE (0x00898, RW) 325
4.4.3.4 Flow Control Registers Description 326
4.4.3.4.1 Priority Flow Control Type Opcode – PFCTOP (0x03008; RW) 326
4.4.3.4.2 Flow Control Transmit Timer Value n – FCTTVn (0x03200 + 4*n[n=0..3]; RW) 326
4.4.3.4.3 Flow Control Receive Threshold Low – FCRTL (0x03220 + 8*n[n=0..7]; RW) 326
4.4.3.4.4 Flow Control Receive Threshold High – FCRTH (0x03260 + 8*n[n=0..7]; RW) 327
4.4.3.4.5 Flow Control Refresh Threshold Value – FCRTV (0x032A0; RW) 328
4.4.3.4.6 Transmit Flow Control Status – TFCS (0x0CE00; RO) 328
4.4.3.5 Receive DMA Registers 329
4.4.3.5.1 Receive Descriptor Base Address Low – RDBAL (0x01000 + 0x40*n[n=0..63]; RW) 329
4.4.3.5.2 Receive Descriptor Base Address High – RDBAH (0x01004 + 0x40*n[n=0..63]; RW) 329
4.4.3.5.3 Receive Descriptor Length – RDLEN (0x01008 + 0x40*n[n=0..63]; RW) 329
4.4.3.5.4 Receive Descriptor Head – RDH (0x01010 + 0x40*n[n=0..63]; RO) 329
4.4.3.5.5 Receive Descriptor Tail – RDT (0x01018 + 0x40*n[n=0..63]; RW) 330
4.4.3.5.6 Receive Descriptor Control – RXDCTL (0x01028 + 0x40*n[n=0..63]; RW) 330
4.4.3.5.7 Split Receive Control Registers – SRRCTL (0x02100 – 0x0213C; RW) 332
4.4.3.5.8 Rx DCA Control Register – DCA_RXCTRL (0x02200 – 0x0223C; RW) 332
4.4.3.5.9 Receive DMA Control Register – RDRXCTL (0x02F00; RW) 333
4.4.3.5.10 Receive Packet Buffer Size – RXPBSIZE (0x03C00 – 0x03C1C; RW) 334
4.4.3.5.11 Receive Control Register – RXCTRL (0x03000; RW) 334
4.4.3.5.12 Drop Enable Control – DROPEN (0x03D04 – 0x03D08; RW) 335
4.4.3.6 Receive Registers 335
4.4.3.6.1 Receive Checksum Control – RXCSUM (0x05000; RW) 335
4.4.3.6.2 Receive Filter Control Register – RFCTL (0x05008; RW) 336
4.4.3.6.3 Multicast Table Array – MTA (0x05200-0x053FC; RW) 336

4.4.3.6.4 Receive Address Low – RAL (0x05400 + 8*n[n=0..15]; RW) 337
4.4.3.6.5 Receive Address High – RAH (0x05404 + 8*n[n=0..15]; RW) 337
4.4.3.6.6 Packet Split Receive Type Register – PSRTYPE (0x05480 – 0x054BC, RW) 338
4.4.3.6.7 VLAN Filter Table Array – VFTA (0x0A000-0x0A9FC; RW) 339
4.4.3.6.8 Filter Control Register – FCTRL (0x05080, RW) 341
4.4.3.6.9 VLAN Control Register – VLNCTRL (0x05088, RW) 343
4.4.3.6.10 Multicast Control Register – MCSTCTRL (0x05090, RW) 343
4.4.3.6.11 Multiple Receive Queues Command Register MRQC (0x05818; RW) 343
4.4.3.6.12 VMDq Control Register – VMD_CTL (0x0581C; RW) 344
4.4.3.6.13 Immediate Interrupt Rx IMIR (0x05A80 + 4*n[n=0..7], RW) 344
4.4.3.6.14 Immediate Interrupt Rx Extended IMIREXT (0x05AA0 + 4*n[n=0..7], RW) 345
4.4.3.6.15 Immediate Interrupt Rx VLAN Priority Register IMIRVP (0x05AC0, RW) 345
4.4.3.6.16 Indirection Table – RETA (0x05C00-0x0057C; RW) 346
4.4.3.6.17 RSS Random Key Register – RSSRK (0x05C80-0x05CA4; RW) 347
4.4.3.7 Transmit Register Descriptions 347
4.4.3.7.1 Transmit Descriptor Base Address Low – TDBAL (0x06000 + n*0x40[n=0..31]; RW) 347
4.4.3.7.2 Transmit Descriptor Base Address High – TDBAH (0x06004 + n*0x40[n=0..31]; RW) 348
4.4.3.7.3 Transmit Descriptor Length – TDLEN (0x06008 + n*0x40[n=0..31]; RW) 348
4.4.3.7.4 Transmit Descriptor Head – TDH (0x06010 + n*0x40[n=0..31]; RO) 348
4.4.3.7.5 Transmit Descriptor Tail – TDT (0x06018 + n*0x40[n=0..31]; RW) 348
4.4.3.7.6 Transmit Descriptor Control – TXDCTL (0x06028 + n*0x40[n=0..31]; RW) 349
4.4.3.7.7 Tx Descriptor Completion Write Back Address Low – TDWBAL (0x06038 + n*0x40[n=0..31]; RW) 350
4.4.3.7.8 Tx Descriptor Completion Write Back Address High – TDWBAH (0x0603C + n*0x40[n=0..31]; RW) 350
4.4.3.7.9 DMA TX Control – DTXCTL (0x07E00; RW) 350
4.4.3.7.10 Tx DCA Control Register – DCA_TXCTRL (0x07200 – 0x0723C; RW) 351
4.4.3.7.11 Transmit IPG Control – TIPG (0x0CB00; RW) 351
4.4.3.7.12 Transmit Packet Buffer Size – TXPBSIZE (0x0CC00 – 0x0CC1C; RW) 352
4.4.3.7.13 Manageability Transmit TC Mapping – MNGTXMAP (0x0CD10; RW) 352
4.4.3.8 Wake-Up Control Registers 352
4.4.3.8.1 Wake Up Control Register – WUC (0x05800; RW) 352
4.4.3.8.2 Wake Up Filter Control Register – WUFC (0x05808; RW) 353
4.4.3.8.3 Wake Up Status Register – WUS (0x05810; RO) 353
4.4.3.8.4 IP Address Valid – IPAV (0x5838; RW) 354
4.4.3.8.5 IPv4 Address Table – IP4AT (0x05840 + n*8 [n = 0..3]; RW) 355
4.4.3.8.6 IPv6 Address Table – IP6AT (0x05880-0x0588C; RW) 355
4.4.3.8.7 Wake Up Packet Length – WUPL (0x05900; R) 356
4.4.3.8.8 Wake Up Packet Memory (128 Bytes) – WUPM (0x05A00-0x05A7C; R) 356
4.4.3.8.9 Flexible Host Filter Table Registers – FHFT (0x09000 – 0x093FC; RW) 356
4.4.3.9 Statistic Registers 357
4.4.3.9.1 CRC Error Count – CRCERRS (0x04000; R) 358
4.4.3.9.2 Illegal Byte Error Count – ILLERRC (0x04004; R) 358
4.4.3.9.3 Error Byte Count – ERRBC (0x04008; R) 358
4.4.3.9.4 MAC Short Packet Discard Count – MSPDC (0x04010; R) 358
4.4.3.9.5 Missed Packets Count – MPC (0x03FA0 – 0x03FBC; R) 359
4.4.3.9.6 MAC Local Fault Count – MLFC (0x04034; R) 359
4.4.3.9.7 MAC Remote Fault Count – MRFC (0x04038; R) 359
4.4.3.9.8 Receive Length Error Count – RLEC (0x04040; R) 359
4.4.3.9.9 Link XON Transmitted Count – LXONTXC (0x03F60; R) 359
4.4.3.9.10 Link XON Received Count – LXONRXC (0x0CF60; R) 360
4.4.3.9.11 Link XOFF Transmitted Count – LXOFFTXC (0x03F68; R) 360
4.4.3.9.12 Link XOFF Received Count – LXOFFRXC (0x0CF68; R) 360
4.4.3.9.13 Priority XON Transmitted Count – PXONTXC (0x03F00 – 0x03F1C; R) 360
4.4.3.9.14 Priority XON Received Count – PXONRXC (0x0CF00 – 0x0CF1C; R) 360
4.4.3.9.15 Priority XOFF Transmitted Count – PXOFFTXC (0x03F20 – 0x03F3C; R) 361
4.4.3.9.16 Priority XOFF Received Count – PXOFFRXC (0x0CF20 – 0x0CF2C; R) 361
4.4.3.9.17 Packets Received (64 Bytes) Count – PRC64 (0x0405C; R) 361

4.4.3.9.18 Packets Received (65-127 Bytes) Count – PRC127 (0x04060; R) 361
4.4.3.9.19 Packets Received (128-255 Bytes) Count – PRC255 (0x04064; R)..... 362
4.4.3.9.20 Packets Received (256-511 Bytes) Count – PRC511 (0x04068; R)..... 362
4.4.3.9.21 Packets Received (512-1023 Bytes) Count – PRC1023 (0x0406C; R)..... 362
4.4.3.9.22 Packets Received (1024 to Max Bytes) Count – PRC1522 (0x04070; R) 362
4.4.3.9.23 Good Packets Received Count – GPRC (0x04074; R) 363
4.4.3.9.24 Broadcast Packets Received Count – BPRC (0x04078; R) 363
4.4.3.9.25 Multicast Packets Received Count – MPRC (0x0407C; R)..... 363
4.4.3.9.26 Good Packets Transmitted Count – GPTC (0x04080; R) 363
4.4.3.9.27 Good Octets Received Count – GORC (0x0408C; R) 364
4.4.3.9.28 Good Octets Transmitted Count – GOTC (0x04094; R); 364
4.4.3.9.29 Receive No Buffers Count – RNBC (0x03FC0 – 0x03FDC; R)..... 364
4.4.3.9.30 Receive Undersize Count – RUC (0x040A4; R)..... 365
4.4.3.9.31 Receive Fragment Count – RFC (0x040A8; R)..... 365
4.4.3.9.32 Receive Oversize Count – ROC (0x040AC; R) 365
4.4.3.9.33 Receive Jabber Count – RJC (0x040B0; R)..... 365
4.4.3.9.34 Management Packets Received Count – MNGPRC (0x040B4; R) 366
4.4.3.9.35 Management Packets Dropped Count – MNGPDC (0x040B8; R)..... 366
4.4.3.9.36 Management Packets Transmitted Count – MNGPTC (0x0CF90; R) 366
4.4.3.9.37 Total Octets Received – TOR (0x040C4; R);..... 366
4.4.3.9.38 Total Packets Received – TPR (0x040D0; R)..... 367
4.4.3.9.39 Total Packets Transmitted – TPT (0x040D4; R)..... 367
4.4.3.9.40 Packets Transmitted (64 Bytes) Count – PTC64 (0x040D8; R) 367
4.4.3.9.41 Packets Transmitted (65-127 Bytes) Count – PTC127 (0x040DC; R)..... 367
4.4.3.9.42 Packets Transmitted (128-255 Bytes) Count – PTC255 (0x040E0; R)..... 368
4.4.3.9.43 Packets Transmitted (256-511 Bytes) Count – PTC511 (0x040E4; R)..... 368
4.4.3.9.44 Packets Transmitted (512-1023 Bytes) Count – PTC1023 (0x040E8; R) 368
4.4.3.9.45 Packets Transmitted (Greater than 1024 Bytes) Count – PTC1522 (0x040EC; R)368
4.4.3.9.46 Multicast Packets Transmitted Count – MPTC (0x040F0; R) 369
4.4.3.9.47 Broadcast Packets Transmitted Count – BPTC (0x040F4; R) 369
4.4.3.9.48 XSUM Error Count – XEC (0x04120; RO) 369
4.4.3.9.49 Receive Queue Statistic Mapping Registers RQSMR (0x2300 + 4*n [n=0…15], RW) 369
4.4.3.9.50 Transmit Queue Statistic Mapping Registers TQSMR (0x7300 + 4*n [n=0…7], RW) 370
4.4.3.9.51 Queue Packets Received Count – QPRC (0x01030+ n*0x40[n=0..15]; R) 371
4.4.3.9.52 Queue Packets Transmitted Count – QPTC (0x06030 + n*0x40[n=0..15]; R) .. 371
4.4.3.9.53 Queue Bytes Received Count – QBRC (0x1034 + n*0x40[n=0..15]; R)..... 371
4.4.3.9.54 Queue Bytes Transmitted Count – QBTC (0x6034+n*0x40[n=0..15]; R)..... 372
4.4.3.10 Management Filter Registers 372
4.4.3.10.1 Management VLAN TAG Value – MAVTV (0x5010 +4*n[n=0..7]; RW)..... 372
4.4.3.10.2 Management Flex UDP/TCP Ports – MFUTP (0x5030 + 4*n[n=0..7]; RW)..... 372
4.4.3.10.3 Management Control Register – MANC (0x05820; RW) 372
4.4.3.10.4 Manageability Filters Valid – MFVAL (0x5824; RW)..... 373
4.4.3.10.5 Management Control To Host Register – MANC2H (0x5860; RW) 374
4.4.3.10.6 Manageability Decision Filters- MDEF (0x5890 + 4*n[n=0..7]; RW)..... 374
4.4.3.10.7 Manageability IP Address Filter – MIPAF (0x58B0-0x58EC; RW)..... 376
4.4.3.10.8 Manageability MAC Address Low – MMAL (0x5910 + 8*n[n=0..3]; RW) 379
4.4.3.10.9 Manageability MAC Address High – MMAH (0x5914 + 8*n[n=0..3]; RW)..... 379
4.4.3.10.10 Flexible TCO Filter Table Registers – FTFT (0x09400-0x097FC; RW) 379
4.4.3.11 PCIe Registers..... 381
4.4.3.11.1 PCIe Control Register – GCR (0x11000; RW)..... 381
4.4.3.11.2 PCIe Timer Value – GTV (0x11004; RW)..... 383
4.4.3.11.3 Function-Tag Register FUNCTAG (0x11008; RW) 383
4.4.3.11.4 PCIe Latency Timer – GLT (0x1100C; RW)..... 384
4.4.3.11.5 Function Active and Power State to Manageability – FACTPS (0x10150; RO).... 384
4.4.3.11.6 PCIe Analog Configuration Register – PCIEANACTL (0x11040; RW) 385
4.4.3.11.7 Software Semaphore Register – SWSM (0x10140; RW) 385
4.4.3.11.8 Firmware Semaphore Register – FWSM (0x10148; RW)..... 386
4.4.3.11.9 General Software Semaphore Register – GSSR (0x10160; RW)..... 387

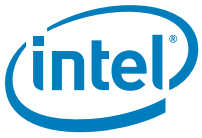


4.4.3.11.10 Mirrored Revision ID- MREVID (0x11064; RO) 390
4.4.3.12 DCA Control Registers 390
4.4.3.12.1 DCA Requester ID Information Register- DCA_ID (0x11070; R) 390
4.4.3.12.2 DCA Control Register- DCA_CTRL (0x11074; RW) 390
4.4.3.13 MAC Registers 391
4.4.3.13.1 PCS_1G Global Config Register 1 – PCS1GCFIG (0x04200, RW) 391
4.4.3.13.2 PCG_1G Link Control Register – PCS1GLCTL (0x04208; RW) 391
4.4.3.13.3 PCS_1G Link Status Register – PCS1GLSTA (0x0420C; RO) 392
4.4.3.13.4 PCS_1 Gb/s Auto Negotiation Advanced Register PCS1GANA (0x04218; RW) ... 393
4.4.3.13.5 PCS_1GAN LP Ability Register – PCS1GANLP (0x0421C; RO) 393
4.4.3.13.6 PCS_1G Auto Negotiation Next Page Transmit Register – PCS1GANNP (0x04220; RW) 394
4.4.3.13.7 PCS_1G Auto Negotiation LP's Next Page Register – PCS1GANLPNP (0x04224; RO) 395
4.4.3.13.8 Flow Control 0 Register – HLREG0 (0x04240, RW) 396
4.4.3.13.9 Flow Control Status 1 Register- HLREG1 (0x04244, RO) 397
4.4.3.13.10 Pause and Pace Register – PAP (0x04248, RW) 398
4.4.3.13.11 MDI Auto-Scan Command and Address – MACA (0x0424C; RW) 399
4.4.3.13.12 Auto-Scan PHY Address Enable – APAE (0x04250; RW) 399
4.4.3.13.13 Auto-Scan Read Data – ARD (0x04254; RW) 399
4.4.3.13.14 Auto-Scan Interrupt Status – AIS (0x04258; RW) 400
4.4.3.13.15 MDI Single Command and Address – MSCA (0x0425C; RW) 400
4.4.3.13.16 MDI Single Read and Write Data – MSRWD (0x04260; RW) 401
4.4.3.13.17 Low MAC Address – MLADD (0x04264; RW) 401
4.4.3.13.18 MAC Address High and Max Frame Size – MHADD (0x04268; RW) 401
4.4.3.13.19 XGXS Status 1 – PCSS1 (0x4288; RO) 401
4.4.3.13.20 XGXS Status 2 – PCSS2 (0x0428C; RO) 402
4.4.3.13.21 10GBASE-X PCS Status – XPCSS (0x04290; RO) 402
4.4.3.13.22 SerDes Interface Control Register – SERDESC (0x04298; RW) 404
4.4.3.13.23 FIFO Status/CNTL Report Register – MACS (0x0429C; RW) 405
4.4.3.13.24 Auto Negotiation Control Register – AUTOC (0x042A0; RW) 405
4.4.3.13.25 Link Status Register – LINKS (0x042A4; RO) 407
4.4.3.13.26 Auto Negotiation Control 2 Register – AUTOC2 (0x042A8; RW) 408
4.4.3.13.27 Auto Negotiation Control 3 Register – AUTOC3 (0x042AC; RW) 409
4.4.3.13.28 Auto Negotiation Link Partner Link Control Word 1 Register – ANLP1 (0x042B0; RO) 409
4.4.3.13.29 Auto Negotiation Link Partner Link Control Word 2 Register – ANLP2 (0x042B4; RO) 410
4.4.3.13.30 MAC Manageability Control Register – MMNGC (0x042D0; Host-RO/MNG-RW) . 410
4.4.3.13.31 Auto Negotiation Link Partner Next Page 1 Register – ANLPNP1 (0x042D4; RO) 411
4.4.3.13.32 Auto Negotiation Link Partner Next Page 2 Register – ANLPNP2 (0x042D8; RO) 411
4.4.3.13.33 Core Analog Configuration Register - ATLASCTL (0x04800; RW) 411

5. System Manageability 413

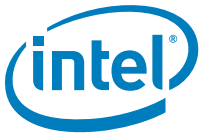
5.1 Pass-Through (PT) Functionality 413
5.2 Components of a Sideband Interface 414
5.3 SMBus Pass-Through Interface 415
5.3.1 General 415
5.3.2 Pass-Through Capabilities 415
5.3.2.1 Packet Filtering 415
5.3.3 Pass-Through Multi-Port Modes 416
5.3.3.1 Automatic Ethernet ARP Operation 416
5.3.3.2 Manageability Receive Filtering 416
5.3.3.2.1 Overview and General Structure 416
5.3.3.2.2 L2 Layer Filtering 418
5.3.3.2.2.1 VLAN Filtering 419
5.3.3.2.3 Manageability Decision Filtering 420
5.3.3.2.3.1 L3 & L4 Filters 421
5.3.3.2.4 Manageability Decision Filters 422
5.3.4 SMBus Transactions 425

5.3.4.1 SMBus Addressing 426
5.3.4.2 SMBus ARP Functionality 426
5.3.4.2.1 SMBus ARP Flow 426
5.3.4.2.2 SMBus ARP UDID Content 428
5.3.4.2.3 SMBus ARP in Dual/Single Mode 430
5.3.4.3 Concurrent SMBus Transactions 430
5.3.5 SMBus Notification Methods 430
5.3.5.1 SMBus Alert and Alert Response Method 431
5.3.5.2 Asynchronous Notify Method 432
5.3.5.3 Direct Receive Method 432
5.3.6 1 MHz SMBus Support 433
5.3.7 Receive TCO Flow 433
5.3.8 Transmit TCO Flow 433
5.3.8.1 Transmit Errors in Sequence Handling 434
5.3.8.2 TCO Command Aborted Flow 435
5.3.9 SMBus ARP Transactions 435
5.3.9.1 Prepare to ARP 435
5.3.9.2 Reset Device (General) 435
5.3.9.3 Reset Device (Directed) 436
5.3.9.4 Assign Address 436
5.3.9.5 Get UDID (General and Directed) 437
5.3.10 SMBus Pass-Through Transactions 439
5.3.10.1 Write Transactions 439
5.3.10.1.1 Transmit Packet Command 439
5.3.10.1.2 Request Status Command 439
5.3.10.1.3 Receive Enable Command 440
5.3.10.1.3.1 Management MAC Address (Data Bytes 7:2) 442
5.3.10.1.3.2 Management IP Address (Data Bytes 11:8) 442
5.3.10.1.3.3 Asynchronous Notification SMBus Address (Data Byte 12) 442
5.3.10.1.3.4 Interface Data (Data Byte 13) 442
5.3.10.1.3.5 Alert Value Data (Data Byte 14) 442
5.3.10.1.4 Force TCO Command 442
5.3.10.1.5 Management Control 443
5.3.10.1.6 Update Management Receive Filter Parameters 444
5.3.10.2 Read Transactions (82598 to BMC) 446
5.3.10.2.1 Receive TCO LAN Packet Transaction 447
5.3.10.2.1.1 Receive TCO LAN Status Payload Transaction 448
5.3.10.2.2 Read Status Command 451
5.3.10.2.3 Get System MAC Address 455
5.3.10.2.4 Read Configuration 455
5.3.10.2.5 Read Management Parameters 456
5.3.10.2.6 Read Management Receive Filter Parameters 456
5.3.10.2.7 Read Receive Enable Configuration 458
5.3.11 LAN Fail-Over in LAN Teaming Mode 459
5.3.11.1 Fail-Over Functionality 459
5.3.11.1.1 Transmit Functionality 459
5.3.11.1.2 Receive Functionality 459
5.3.11.1.3 Port Switching (Fail-Over) 459
5.3.11.1.4 Device Driver Interactions 460
5.3.11.2 Fail-Over Configuration 460
5.3.11.2.1 Preferred Primary Port 460
5.3.11.2.2 Gratuitous ARPs 460
5.3.11.2.3 Link Down Timeout 460
5.3.11.3 Fail-Over Register 460
5.3.12 SMBus Troubleshooting Guide 461
5.3.12.1 TCO Alert Line Stays Asserted After a Power Cycle 462
5.3.12.2 SMBus Commands are Always NACK'd by the 82598 462
5.3.12.3 SMBus Clock Speed is 16.6666 KHz 462
5.3.12.4 A Network Based Host Application is not Receiving any Network Packets 463
5.3.12.5 Status Registers 463



5.3.12.5.1 Firmware Semaphore Register (FWSM, 0x10148) 463
5.3.12.5.2 Management Control Register (MANC 0x5820) 463
5.3.12.5.3 Management Control To Host Register (MANC2H 0x5860) 464
5.3.12.6 Unable to Transmit Packets from the BMC 464
5.3.12.7 SMBus Fragment Size 464
5.3.12.8 Enable XSum Filtering 465
5.3.12.9 Still Having Problems? 465
5.3.13 Sample Configurations 465
5.4 NC-SI Interface 475
5.4.1 Overview 475
5.4.1.1 Terminology 475
5.4.1.2 System Topology 477
5.4.1.3 Data Transport 478
5.4.1.3.1 Control Frames 478
5.4.1.3.2 NC-SI Frames Receive Flow 479
5.4.2 NC-SI Support 479
5.4.2.1 Supported Features 479
5.4.2.2 NC-SI Mode - Intel Specific Commands 484
5.4.2.2.1 Overview 484
5.4.2.2.1.1 OEM Command (0x50) 485
5.4.2.2.1.2 OEM Response (0xD0) 485
5.4.2.2.1.3 OEM Specific Command Response Reason Codes 486
5.4.2.2.2 Proprietary Commands Format 487
5.4.2.2.2.1 Set Intel Filters Control Command (Intel Command 0x00) 487
5.4.2.2.2.2 Set Intel Filters Control Response Format (Intel Command 0x00) 488
5.4.2.2.3 Set Intel Filters Control - IP Filters Control Command (Intel Command 0x00, Filter Control Index 0x00) 488
5.4.2.2.3.1 Set Intel Filters Control - IP Filters Control Response (Intel Command 0x00, Filter Control Index 0x00) 489
5.4.2.2.4 Get Intel Filters Control Command (Intel Command 0x01) 489
5.4.2.2.4.1 Get Intel Filters Control - IP Filters Control Command (Intel Command 0x01, Filter Control Index 0x00) 489
5.4.2.2.4.2 Get Intel Filters Control - IP Filters Control Response (Intel Command 0x01, Filter Control Index 0x00) 490
5.4.2.2.5 Set Intel Filters Formats 490
5.4.2.2.5.1 Set Intel Filters Command (Intel Command 0x02) 490
5.4.2.2.5.2 Set Intel Filters Response (Intel Command 0x02) 490
5.4.2.2.5.3 Set Intel Filters - Manageability to Host Command (Intel Command 0x02, Filter Parameter 0x0A) 491
5.4.2.2.5.4 Set Intel Filters - Manageability to Host Response (Intel Command 0x02, Filter Parameter 0x0A) 491
5.4.2.2.5.5 Set Intel Filters - Flex Filter 0 Enable Mask and Length Command (Intel Command 0x02, Filter Parameter 0x10/0x20/0x30/0x40) 492
5.4.2.2.5.6 Set Intel Filters - Flex Filter 0 Enable Mask and Length Response (Intel Command 0x02, Filter Parameter 0x10/0x20/0x30/0x40) 492
5.4.2.2.5.7 Set Intel Filters - Flex Filter 0 Data Command (Intel Command 0x02, Filter Parameter 0x11/0x21/0x31/0x41) 493
5.4.2.2.5.8 Set Intel Filters - Flex Filter 0 Data Response (Intel Command 0x02, Filter Parameter 0x11/0x21/0x31/0x41) 493
5.4.2.2.5.9 Set Intel Filters - Packet Addition Decision Filter Command (Intel Command 0x02, Filter Parameter 0x61) 493
5.4.2.2.5.10 Set Intel Filters - Packet Addition Decision Filter Response (Intel Command 0x02, Filter Parameter 0x61) 495
5.4.2.2.5.11 Set Intel Filters - Flex TCP/UDP Port Filter Command (Intel Command 0x02, Filter Parameter 0x63) 495
5.4.2.2.5.12 Set Intel Filters - Flex TCP/UDP Port Filter Response (Intel Command 0x02, Filter Parameter 0x63) 496
5.4.2.2.5.13 Set Intel Filters - IPv4 Filter Command (Intel Command 0x02, Filter Parameter 0x64) 496

5.4.2.2.5.14 Set Intel Filters - IPv4 Filter Response (Intel Command 0x02, Filter Parameter 0x64) 496
5.4.2.2.5.15 Set Intel Filters - IPv6 Filter Command (Intel Command 0x02, Filter Parameter 0x65) 497
5.4.2.2.5.16 Set Intel Filters - IPv6 Filter Response (Intel Command 0x02, Filter Parameter 0x65) 498
5.4.2.2.6 Get Intel Filters Formats 498
5.4.2.2.6.1 Get Intel Filters Command (Intel Command 0x03) 498
5.4.2.2.6.2 Get Intel Filters Response (Intel Command 0x03) 498
5.4.2.2.6.3 Get Intel Filters - Manageability to Host Command (Intel Command 0x03, Filter Parameter 0x0A) 498
5.4.2.2.6.4 Get Intel Filters - Manageability to Host Response (Intel Command 0x03, Filter Parameter 0x0A) 499
5.4.2.2.6.5 Get Intel Filters - Flex Filter 0 Enable Mask and Length Command (Intel Command 0x03, Filter Parameter 0x10/0x20/0x30/0x40) 499
5.4.2.2.6.6 Get Intel Filters - Flex Filter 0 Enable Mask and Length Response (Intel Command 0x03, Filter Parameter 0x10/0x20/0x30/0x40) 500
5.4.2.2.6.7 Get Intel Filters - Flex Filter 0 Data Command (Intel Command 0x03, Filter Parameter 0x11/0x21/0x31/0x41) 500
5.4.2.2.6.8 Get Intel Filters - Flex Filter 0 Data Response (Intel Command 0x03, Filter Parameter 0x11) 501
5.4.2.2.6.9 Get Intel Filters - Packet Addition Decision Filter Command (Intel Command 0x03, Filter Parameter 0x61) 501
5.4.2.2.6.10 Get Intel Filters - Packet Addition Decision Filter Response (Intel Command 0x03, Filter Parameter 0x0A) 502
5.4.2.2.6.11 Get Intel Filters - Flex TCP/UDP Port Filter Command (Intel Command 0x03, Filter Parameter 0x63) 502
5.4.2.2.6.12 Get Intel Filters - Flex TCP/UDP Port Filter Response (Intel Command 0x03, Filter Parameter 0x63) 502
5.4.2.2.6.13 Get Intel Filters - IPv4 Filter Command (Intel Command 0x03, Filter Parameter 0x64) 503
5.4.2.2.6.14 Get Intel Filters - IPv4 Filter Response (Intel Command 0x03, Filter Parameter 0x64) 503
5.4.2.2.6.15 Get Intel Filters - IPv6 Filter Command (Intel Command 0x03, Filter Parameter 0x65) 503
5.4.2.2.6.16 Get Intel Filters - IPv6 Filter Response (Intel Command 0x03, Filter parameter 0x65) 504
5.4.2.2.7 Set Intel Packet Reduction Filters Formats 504
5.4.2.2.7.1 Set Intel Packet Reduction Filters Command (Intel Command 0x04) 504
5.4.2.2.7.2 Set Intel Packet Reduction Filters Response (Intel Command 0x04) 504
5.4.2.2.7.3 Set Unicast Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x00) 505
5.4.2.2.7.4 Set Unicast Packet Reduction Response (Intel Command 0x04, Reduction Filter Index 0x00) 507
5.4.2.2.7.5 Set Multicast Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x01) 507
5.4.2.2.7.6 Set Multicast Packet Reduction Response (Intel Command 0x04, Reduction Filter Index 0x01) 509
5.4.2.2.7.7 Set Broadcast Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x02) 509
5.4.2.2.7.8 Set Broadcast Packet Reduction Response (Intel Command 0x08) 511
5.4.2.2.8 Get Intel Packet Reduction Filters Formats 511
5.4.2.2.8.1 Get Intel Packet Reduction Filters Command (Intel Command 0x05) 511
5.4.2.2.8.2 Set Intel Packet Reduction Filters Response (Intel Command 0x05) 512
5.4.2.2.8.3 Get Unicast Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x00) 512
5.4.2.2.8.4 Get Unicast Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x00) 512
5.4.2.2.8.5 Get Multicast Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x01) 513



5.4.2.2.8.6 Get Multicast Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x01) 513
5.4.2.2.8.7 Get Broadcast Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x02) 513
5.4.2.2.8.8 Get Broadcast Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x02) 514
5.4.2.2.9 System MAC Address 514
5.4.2.2.9.1 Get System MAC Address Command (Intel Command 0x06) 514
5.4.2.2.9.2 Get System MAC Address Response (Intel Command 0x06) 514
5.4.2.2.10 Set Intel Management Control Formats 515
5.4.2.2.10.1 Set Intel Management Control Command (Intel Command 0x20) 515
5.4.2.2.10.2 Set Intel Management Control Response (Intel Command 0x20) 515
5.4.2.2.11 Get Intel Management Control Formats 516
5.4.2.2.11.1 Get Intel Management Control Command (Intel Command 0x21) 516
5.4.2.2.11.2 Get Intel Management Control Response (Intel Command 0x21) 516
5.4.2.2.12 TCO Reset 516
5.4.2.2.12.1 Perform Intel TCO Reset Command (Intel Command 0x22) 516
5.4.2.2.12.2 Perform Intel TCO Reset Response (Intel Command 0x22) 517
5.4.2.2.13 Checksum Offloading 517
5.4.2.2.13.1 Enable Checksum Offloading Command (Intel Command 0x23) 517
5.4.2.2.13.2 Enable Checksum Offloading Response (Intel Command 0x23) 517
5.4.2.2.13.3 Disable Checksum Offloading Command (Intel Command 0x24) 518
5.4.2.2.13.4 Disable Checksum Offloading Response (Intel Command 0x24) 518
5.4.3 Basic NC-SI Workflows 518
5.4.3.1 Package States 518
5.4.3.2 Channel States 518
5.4.3.3 Discovery 519
5.4.3.4 Configurations 519
5.4.3.4.1 NC Capabilities Advertisement 519
5.4.3.4.2 Receive Filtering 519
5.4.3.4.2.1 MAC Address Filtering 520
5.4.3.4.2.2 VLAN 520
5.4.3.5 Pass-Through Traffic States 521
5.4.3.5.1 Channel Enable 521
5.4.3.5.2 Network Transmit Enable 521
5.4.3.6 Asynchronous Event Notifications 521
5.4.3.7 Querying Active Parameters 522
5.4.4 Resets 522
5.4.5 Advanced Workflows 522
5.4.5.1 Multi-NC Arbitration 522
5.4.5.1.1 Package Selection Sequence Example 523
5.4.5.2 External Link Control 524
5.4.5.2.1 Set Link While LAN PCIe Functionality is Disabled 524
5.4.5.3 Multiple Channels (Fail-Over) 524
5.4.5.3.1 Fail-Over Algorithm Example 525
5.4.5.4 Statistics 525

6 Mechanical Specification 527

6.1 Package Information 527

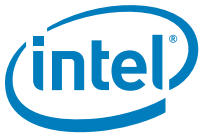
7 Electrical Specifications 529

7.1 Operating Conditions 529
7.2 Absolute Maximum Ratings 529
7.3 Recommended Operating Conditions 530
7.4 Power Delivery 530
7.4.1 Power Supply Specifications 530
7.4.2 Power Supply Sequencing 532
7.4.3 Power Consumption 534
7.5 DC Specifications 536
7.5.1 Digital I/O 536

7.5.2 Open Drain I/O 537
7.5.3 NC-SI I/O 537
7.6 Digital I/F AC Specifications 538
7.6.1 Digital I/O AC Specification 538
7.6.2 EEPROM AC Specifications 540
7.6.3 Flash AC Specification 541
7.6.4 SMBus AC Specification 544
7.6.5 NC-SI AC Specification 545
7.6.6 Reset Signals 547
7.6.6.1 POR_BYPASS (External) 548
7.6.7 PCIe DC/AC Specification 548
7.6.7.1 PCIe Specification (Receiver and Transmitter) 548
7.6.7.2 PCIe Specification (Input Clock) 548
7.6.8 Reference Clock Specification 548

8 Design Guidelines 551

8.1 Connecting the PCIe interface 551
8.1.1 Link Width Configuration 551
8.1.2 Polarity Inversion and Lane Reversal 551
8.1.3 PCIe Reference Clock 551
8.1.4 Bias Resistor 552
8.1.5 Miscellaneous PCIe Signals 552
8.2 Connecting the MAUI Interfaces 552
8.3 MAUI Channels Lane Connections 552
8.3.1 Bias Resistor 552
8.3.2 XAUI, KX/KX4, CX4 and BX Layout Recommendations 552
8.3.2.1 Board Stack Up Example 552
8.3.2.2 Trace Geometries 553
8.3.2.3 Other High-Speed Signal Routing Practices 554
8.3.2.4 Via Usage 555
8.3.2.5 Reference Planes 556
8.3.2.6 Dielectric Weave Compensation 557
8.3.2.7 Impedance Discontinuities 557
8.3.2.8 Reducing Circuit Inductance 557
8.3.2.9 Signal Isolation 557
8.3.2.10 Power and Ground Planes 558
8.4 Connecting the Serial EEPROM 558
8.4.1 Supported EEPROM devices 558
8.4.2 EEUPDATE 559
8.5 Connecting the Flash 559
8.5.1 Supported EEPROM Devices 559
8.6 Connecting the Manageability Interfaces 560
8.6.1 Connecting the SMBus Interface 560
8.6.2 Connecting the NC-SI Interface 560
8.6.3 NC-SI Electrical Interface Requirements 561
8.6.3.1 External Baseboard Management Controller (BMC) 561
8.6.3.2 NC-SI Reference Schematic 561
8.7 Resets 563
8.8 NC-SI Layout Requirements 563
8.8.1 Board Impedance 563
8.8.2 Trace Length Restrictions 563
8.8.3 Special Delay Requirements 565
8.9 Connecting the MDIO Interfaces 565
8.10 Connecting the Software-Definable Pins (SDPs) 565
8.11 Connecting the Light Emitting Diodes for Designs Based on the 82598 Controller 566
8.12 Connecting the Miscellaneous Signals 566
8.12.1 LAN Disable 566
8.12.2 BIOS Handling of Device Disable 568
8.12.3 PHY Disable and Device Power Down Signals 568
8.13 Oscillator Design Considerations 568



8.13.1 Oscillator Types 568
8.13.1.1 Fixed Crystal Oscillator 568
8.13.1.2 Programmable Crystal Oscillators 569
8.13.2 Oscillator Solution 569
8.13.3 Oscillator Layout Recommendations 570
8.13.4 Reference Clock Measurement Recommendations 570
8.14 Power Supplies 570
8.14.1 Power Supply Sequencing 570
8.14.1.1 Using Regulators With Enable Pins 571
8.14.2 Power Supply Filtering 571
8.14.3 Support for Power Management and Wake Up 571
8.15 Connecting the JTAG Port 572
8.16 Thermal Design Considerations 572
8.16.1 Importance of Thermal Management 572
8.16.2 Packaging Terminology 572
8.16.3 Thermal Specifications 573
8.16.3.1 Case Temperature 574
8.16.4 Thermal Attributes 574
8.16.4.1 Typical System Definition 574
8.16.4.2 Package Mechanical Attributes 574
8.16.4.3 Package Thermal Characteristics 574
8.16.5 Thermal Enhancements 576
8.16.5.1 Clearances 576
8.16.5.2 Default Enhanced Thermal Solution 577
8.16.5.3 Extruded Heatsinks 577
8.16.5.3.1 Attaching the Extruded Heatsink 578
8.16.5.3.1.1 Clips 578
8.16.5.3.1.2 Thermal Interface (PCM45 Series) 578
8.16.5.4 Thermal Considerations for Board Design 578
8.16.5.4.1 Reliability 579
8.16.5.5 Thermal Interface Management for Heat-Sink Solutions 579
8.16.5.5.1 Bond Line Management 579
8.16.5.5.2 Interface Material Performance 579
8.16.5.5.3 Thermal Resistance of the Material 580
8.16.5.5.4 Wetting/Filling Characteristics of the Material 580
8.16.6 Measurements for Thermal Specifications 580
8.16.6.1 Case Temperature Measurements 580
8.16.6.2 Attaching the Thermocouple (No Heatsink) 580
8.16.6.3 Attaching the Thermocouple (Heatsink) 581
8.16.7 Heatsink and Attach Suppliers 582
8.16.8 PHY Suppliers 582

9. Diagnostic Registers 583

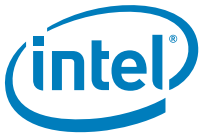
9.1 Register Summary 583
9.1.1 GHOST ECC Register - GHECCR (0x110B0, RW) 583

10. Models, Symbols, Testing Options, Schematics and Checklists 585

10.1 Models and Symbols 585
10.2 Physical Layer Conformance Testing 585
10.3 Schematics 585
10.4 Checklists 585



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



1. General Information

1.1 Introduction

The Intel® 82598 10 GbE Controller is a single, compact, low-power component with two fully integrated Gigabit Ethernet Media Access Control (MAC) and XAUI ports.

The 82598 supports 10GBASE-KX4/1000BASE-KX as in IEEE 802.3ap and CX4 (802.3ak). Ports also contain a serializer-deserializer (designated "BX") to support 1000Base-SX/LX (optical fiber) and GbE backplane applications. CX4 and XAUI interfaces are also supported. In addition to managing MAC and PHY Ethernet layer functions, the controller manages PCIe packet traffic across its transaction, link, and physical/logical layers.

The 82598 supports Intel's Input/Output Acceleration Technology (I/OAT) v2.0. In addition, virtual queues are supported by I/O virtualization.

The 82598's on-board System Management Bus (SMBus) and Network Controller Sideband Interface (NC-SI) ports enable network manageability implementations. With SMBus, management packets can be routed to or from a management processor. SMBus ports enable industry standards, such as Intelligent Platform Management Interface (IPMI). NC-SI ports enable support for the industry DMTF standard.

The 82598, with PCIe architecture, is designed for high-performance and low host-memory access latency. The 82598 connects directly to a system Memory Control Hub (MCH) or I/O Controller Hub (ICH) using one, two, four, or eight PCIe lanes.

Wide internal data paths eliminate performance bottlenecks by handling large address and data words. Combining a parallel and pipelined logic architecture optimized for Ethernet and independent transmit and receive queues, the 82598 efficiently handles packets with minimum latency. The 82598 includes advanced interrupt handling features. It uses efficient ring buffer descriptor data structures, with 32 Tx queues and 64 RX queues. Large on-chip buffers maintain superior performance. In addition, using hardware acceleration, the 82598 offloads tasks from the host, such as TCP/UDP/IP checksum calculations and TCP segmentation.

The 82598 package is a 31 mm x 31 mm, 883-ball, 1.0 mm ball pitch, Flip-Chip Ball Grid Array (FCBGA).

1.2 Terminology and Acronyms

Acronym	Description
ACK	Acknowledge.
ARA	SMBus Alert Response Address.
ARP	Address Resolution Protocol.

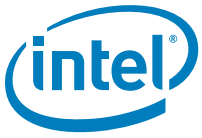
Acronym	Description
ASF	Alert Standard Format. The manageability protocol specification defined by the DMTF.
b/w	Bandwidth.
BMC	Baseboard Manageability Controller. The general name for an external TCO controller, relevant only in TCO Mode.
CML	Current Mode Logic.
CSR	Control and Status Register. Usually refers to a hardware register.
DCA	Direct Cache Access.
DFT	Design for Testability.
DHCP	Dynamic Host Configuration Protocol. A TCP/IP protocol that enables a client to receive a temporary IP address over the network from a remote server.
DMTF	The international organization responsible for managing and maintaining the ASF specification.
FW	Firmware. Also known as embedded software.
GPIO	General Purpose I/O.
HW	Hardware.
IEEE	Institute of Electrical and Electronics Engineers.
IP	Internet Protocol. The protocol within TCP/IP that governs the breakup and reassembly of data messages into packets and the packet routing within the network.
IP Address	The 4-byte or 16-byte address that designates the Ethernet controller within the IP communication protocol. This address is dynamic and can be updated frequently during runtime.
IPC	Inter-Processor Communication.
IPMI	Intelligent Platform Management Interface Specification.
LAN	Local Area Network. Also known as the Ethernet.
LOM	LAN on Motherboard.
MAC	Media Access Controller.
MAC Address	The 6-byte address that designates Ethernet controller within the Ethernet protocol. This address is constant and unique per Ethernet controller.
MAUI	Medium Attachment Unit Interface.
MDIO	Management Data Input/Output Interface.
NA	Not Applicable.
NACK	Not Acknowledged.
NC-SI	Network Controller Sideband Interface.
NIC	Network Interface Card. Generic name for a Ethernet controller that resides on a Printed Circuit Board (PCB).

Acronym	Description
OS	Operating System. Usually designates the PC system's software.
PCS	Physical Coding Sub-Layer.
PEC	The SMBus checksum signature, sent at the end of an SMBus packet. An SMBus device can be configured either to require or not require this signature.
PET	Platform Event Trap.
PHY	Physical Layer Device.
PMA	Physical Medium Attachment.
PMD	Physical Medium Dependent.
PSA	SMBus Persistent Slave Address device. In the SMBus 2.0 specification, this designates an SMBus device whose address is stored in non-volatile memory.
PT	Pass Through. Also known as TCO mode.
RMCP	Remote Management and Control Protocol.
RSP	RMCP Security Extensions Protocol.
SA	Security Association.
SerDes	Serializer And Deserializer Circuit.
SFD	Start Frame Delimiter.
SMBus	System Management Bus.
SNMP	Simple Network Management Protocol.
SW	Software.
TBD	To Be Defined.
TCO	Total Cost of Ownership.
VPD	Vital Product Data (PCI Protocol).
WWDM	Wide Wave Division Multiplexing.
XAUI	10 Gigabit Attachment Unit Interface.
XFP	10 Gigabit Small Form Factor Pluggable Modules.
XGMII	10 Gigabit Media Independent Interface.
XGXS	XGMII Extender Sub-Layer.

1.3 Reference Documents

This application assumes that the designer is acquainted with high-speed design and board layout techniques. The following provide additional information:

- 10GBASE-X – An IEEE 802.3 physical coding sublayer for 10 Gb/s operation over XAUI and four lane PMDs as per IEEE 802.3 Clause 48
- 1000BASE-CX – 1000BASE-X over specialty shielded 150 Ohm balanced copper jumper cable assemblies as specified in IEEE 802.3 Clause 39
- 10GBASE-LX4 – EEE 802.3 Physical Layer specification for 10Gb/s using 10GBASE-X encoding over four WWDM lanes over multimode fiber as specified in IEEE 802.3 Clause 54
- 10GBASE-CX4 – EEE 802.3 Physical Layer specification for 10Gb/s using 10GBASE-X encoding over four lanes of 100 Ohm shielded balanced copper cabling as specified in IEEE 802.3 Clause 54
- 1000BASE-KX – IEEE 802.3 Physical Layer specification for 1Gb/s using 1000BASE-X encoding over an electrical backplane as specified in IEEE 802.3 Clause 70
- 10GBASE-KX4 – IEEE 802.3 Physical Layer specification for 10Gb/s using 10GBASE-X encoding over an electrical backplane as specified in IEEE 802.3 Clause 71
- 10GBASE-KR – IEEE 802.3 Physical Layer specification for 10Gb/s using 10GBASE-R encoding over an electrical backplane as specified in IEEE 802.3 Clause 72
- 1000BASE-BX – 1000BASE-BX is the PICMG 3.1 electrical specification for transmission of 1Gb/s Ethernet or 1Gb/s Fibre Channel encoded data over the backplane
- 10GBASE-BX4 – 10GBASE-BX4 is the PICMG 3.1 electrical specification for transmission of the 10Gb/s XAUI signaling for a backplane environment
- 10GBASE-T – IEEE 802.3 Physical Layer specification for a 10 Gb/s LAN using four pairs of Class E or Class F balanced twisted pair copper cabling as specified in IEEE 802.3 Clause 55
- *IEEE Standard 802.3, 2002 Edition (Ethernet). Incorporates various IEEE Standards previously published separately. Institute of Electrical and Electronic Engineers (IEEE).*
- *IEEE Standard 802.3ap draft D2.2*
- *IEEE Standard 1149.1, 2001 Edition (JTAG). Institute of Electrical and Electronics Engineers (IEEE)*
- *PICMG3.1 Ethernet/Fibre Channel Over PICMG 3.0 Draft Specification January 14, 2003 Version D1.0*
- PCI Express* Specification v2.0 (2.5 GT/s)
- *PCI Specification, version 3.0*
- *IPv4 Specification (RFC 791)*
- *IPv6 Specification (RFC 2460)*
- *TCP/UDP Specification (RFC 793/768)*
- *ARP Specification (RFC 826)*
- *IEEE Standard 802.1Q for VLAN*
- *IETF Internet Draft, Marker PDU Aligned Framing for TCP Specification*
- *IETF Internet Draft, Direct Data Placement over Reliable Transports*
- *System Management Bus (SMBus) Specification, SBS Implementers Forum, Ver. 2.0, August 2000*
- *Advanced Configuration and Power Interface Specification, Rev 2.0b, October 2002*
- *PCI Bus Power Management Interface Specification, Rev. 1.2, March 2004*



- *EUI-64 Specification, http://standards.ieee.org/regauth/oui/tutorials/EUI64.html.*
- *82563EB/82564EB Gigabit Ethernet Physical Layer Device Design Guide, Intel Corporation.*
- System Management Bus BIOS Interface Specification, Revision 1.0. Intel Corporation.
- The I^2C Bus and How to Use It, 1995. Phillips Semiconductors. This document provides electrical and timing specifications for the I^2C busses.
- I^2C Specification v2.1, Phillips Semiconductors
- Intelligent Platform Management Bus (IPMB) Communications Protocol Specification, Version 1.5, 2001, Dell Computer Corporation, Hewlett-Packard Company, Intel Corporation, and NEC Corporation. This document provides the transport protocol, electrical specifications, and specific command specifications for the IPMB.

1.4 Models and Symbols

IBIS, BSDL, and HSPICE modeling files are available from your local Intel representative.

1.5 Physical Layer Conformance Testing

Physical layer conformance testing (also known as IEEE testing) is a fundamental capability for all companies with Ethernet LAN products. If your company does not have the resources and equipment to perform these tests, consider contracting the tests to an outside facility.

Once you integrate an external PHY with the 82598, the electrical performance of the solution should be characterized for conformance.

1.6 Design and Board Layout Checklists

Layout and schematic checklists are available from your local Intel representative.

1.7 Number Conventions

Unless otherwise specified, numbers are represented as follows:

- Hexadecimal numbers are identified by an "h" suffix on the number (2Ah, 12h) or an '0x' prefix.
- Binary numbers are identified by a "b" suffix on the number (0011b). Values for SMBus transactions in diagrams are listed in binary without the "b" or in hexadecimal without the "h"
- Any other numbers without a suffix are intended as decimal numbers.

1.8 System Configurations

The 82598 is designed for systems configured as rack-mounted or pedestal servers where it can be used as an add-on Network Interface Card (NIC) or LAN on Motherboard (LOM). Another system configuration is blade servers, where it can be used as a LOM or on a mezzanine card (see Figure 1-1 and Figure 1-2).

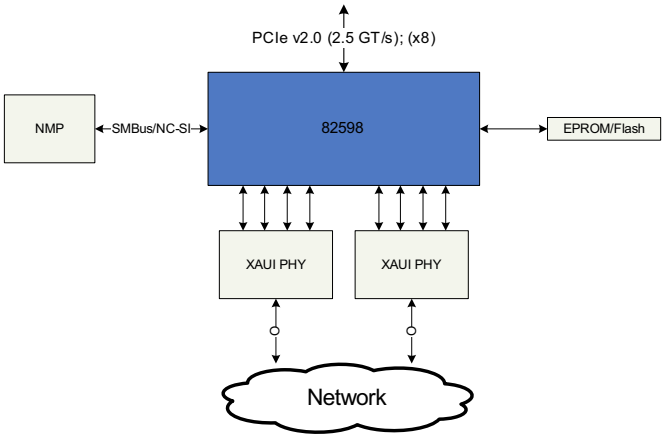


Figure 1-1. Typical NIC System Configuration

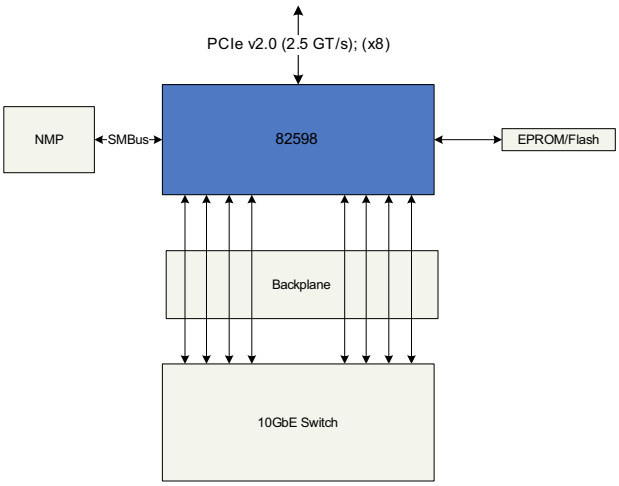


Figure 1-2. Typical Blade System Configuration

1.9 External Interfaces

Figure 1-3 shows the supported 82598 external interfaces.

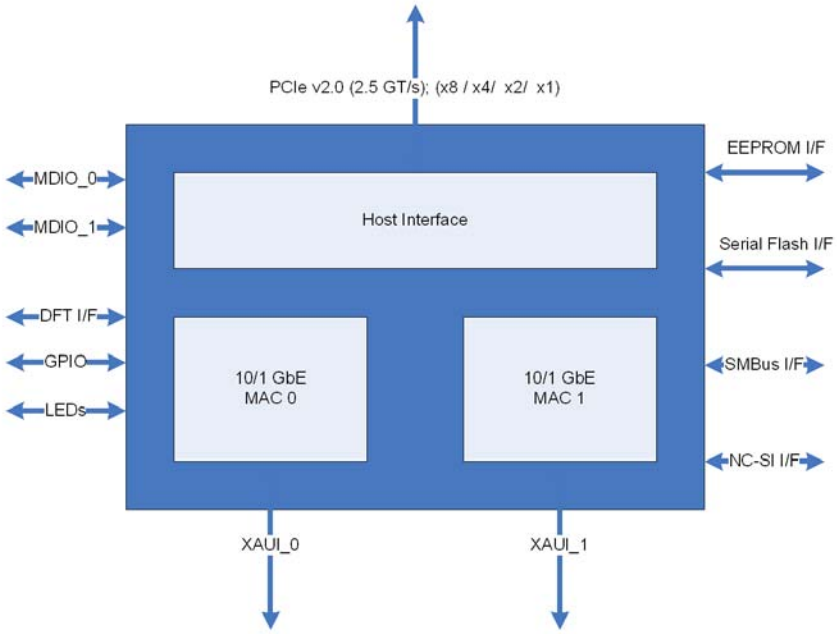


Figure 1-3. Intel® 82598 10 GbE Controller Block Diagram

1.9.1 PCIe Interface

The PCIe v2.0 (2.5 GT/s) interface is used by the 82598 as a host interface. It supports x8, x4, x2 and x1 configurations at a speed of 2.5 GHz. The maximum aggregated raw bandwidth for typical an x8 configuration is 16 Gb/s in each direction. Refer to other sections in this document for a full pin description and interface timing characteristics.

1.9.2 XAUI Interfaces

Two independent XAUI interfaces are used to connect two ports to external devices. They can be configured as an XAUI interface that connects directly to another XAUI compliant device, as a 10GBASE-KX4 interface that connects over a backplane to another KX4 compliant device, or a 10GBASE-CX4 interface that attaches to a CX4 compliant cable.

The 82598 supports IEEE 802.3ae (10 Gb/s) implementations. It performs all of the functions required for transmission and reception handling called out in the standards for an XAUI media interface. It also supports IEEE 802.3ak, IEEE 802.3ap (KX and KX4 only), and PICMG3.1 (BX only) implementations including an auto-negotiation layer and PCS layer synchronization.

The interface can be configured to operate in 1 Gb/s mode of operation (BX and KX). One of the 4 XAUI lanes (lane 0) is used in 1 Gb/s mode.

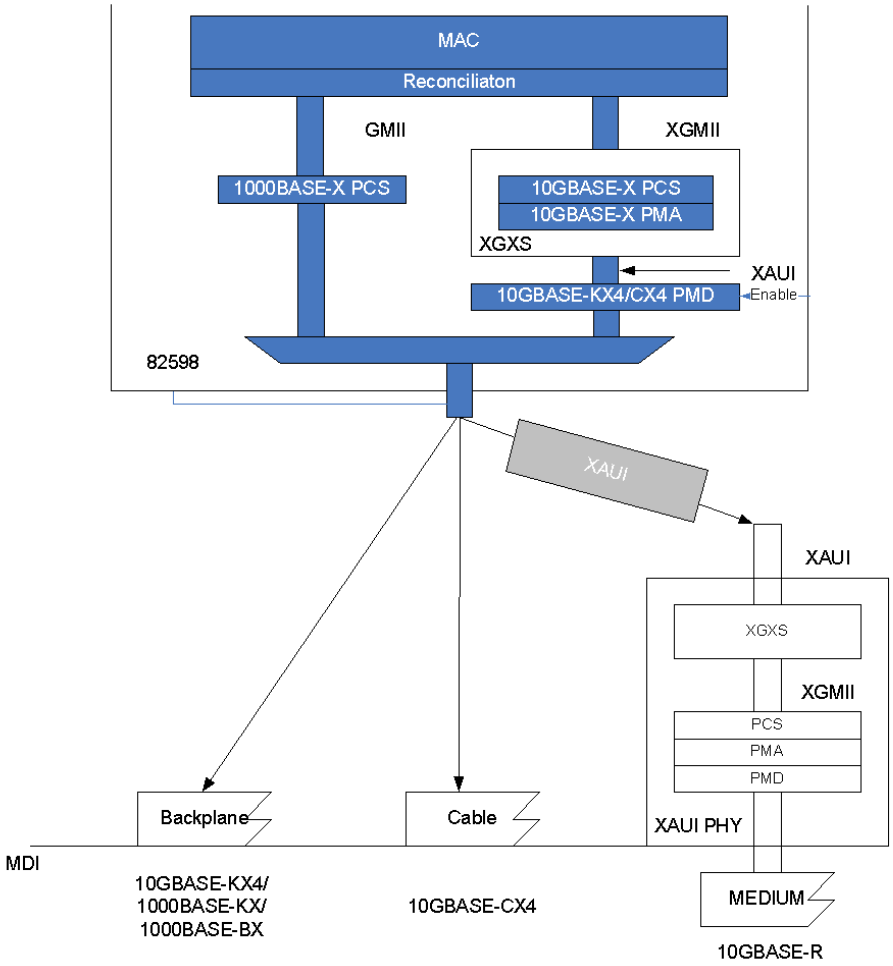


Figure 1-4. Network Interface Connections

Refer to Section 2. for full-pin descriptions and Section 7 for the timing characteristics of those interfaces.

1.9.3 EEPROM Interface

The 82598 uses an EEPROM device for storing product configuration information. Several words of the EEPROM are accessed by the 82598 after reset in order to provide pre-boot configuration data that must be available to it before it is accessed by host software. The remainder of stored information is accessed by various software modules used to report product configuration, serial number, etc.

The 82598 uses a SPI (4-wire) serial EEPROM device such as a AT25040AN or compatible. Refer to Section 2. for full-pin descriptions and Section 7 for the timing characteristics of those interfaces.

1.9.4 Serial Flash Interface

The 82598 provides an external SPI serial interface to a Flash (or boot ROM) device such as the Atmel AT25F1024 or AT25FB512. The 82598 supports serial Flash devices with up to 64 Mb (8 MB) of memory. The size of the Flash used by the 82598 can be configured by the EEPROM.

Note: Though the 82598 supports devices with up to 8 MB of memory, larger devices can be used. Access to memory beyond the Flash device size results in access wrapping as only lower address bits are used by the Flash control unit.

1.9.5 SMBus Interface

SMBus is an optional interface for pass-through and/or configuration traffic between an external BMC and the 82598.

1.9.6 NC-SI Interface

NC-SI is an optional interface for pass-through and/or configuration traffic between a BMC and the 82598.

The following NC-SI capabilities are not supported:

- Collision Detection – The interface supports only full-duplex operation.
- MDIO – MDIO/MDC management traffic is not passed by NC-SI.
- Magic packets – magic packets are not detected by the 82598 NC-SI receive end.
- The 82598 is not 5 V dc tolerant and requires that signals conform to 3.3 V dc signaling.

The NC-SI interface provides a connection to an external BMC and operates in one of the following two modes:

- NC-SI-SMBus Mode – In this mode, the NC-SI interface is functional in conjunction with an SMBus interface, where pass-through traffic passes through NC-SI while configuration traffic passes through SMBus.
- NC-SI Mode – In this mode, the NC-SI interface is functional as a single interface with an external BMC, where all traffic between the 82598 and the BMC flows through this interface.

1.9.7 MDIO Interfaces

The 82598 implements two MII Management Interfaces (also known as the Management Data Input/Output or MDIO Interface) for a control plane connection between the XAUI MAC and PHY devices (master side). This interface provides the MAC and software the ability to monitor and control the state of the PHY. The 82598 supports both 802.3 and 802.3ae data formats for 1 Gb/s and 10 Gb/s operation. The electricals for the MDIO interface are according to 802.3. Those interfaces can be controlled by software via MDI single command and address – MSCA (0x0425C; RW).

Each MDIO interface should be connected to the relevant PHY as shown in the following example (each MDIO interface is driven by the appropriate MAC function).

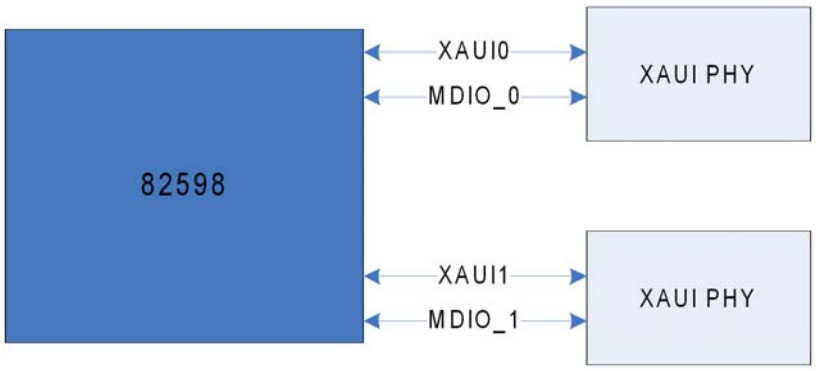


Figure 1-5. MDIO Connection Example

The 82598 MDIO interface is compliant with 802.3 clause 45 (backward compatible to clause 22). However, pin electricals are 3.3 V dc and not 1.2 V dc as defined by clause 45.

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

The 82598 has eight SDP pins per port; these can be used for miscellaneous hardware or software-controllable purposes. Pins can each be individually configurable to act as either input or output pins. The default direction of the lower SDP pins (SDP0[3:0]-SDP1[3:0]) are configurable by EEPROM, as well as the default value of these pins if configured as outputs. To avoid signal contention, all pins are set as input pins until the EEPROM configuration is loaded.

The 82598 also has four of the SDP pins per port; these can be configured for use as General-Purpose Interrupt (GPI) inputs. To act as GPI pins, the pins must be configured as inputs. A corresponding GPI interrupt-detection enable bit 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 Extended SDP Control (ESDP) register and Extended OD SDP Control (EODSDP) register.

1.9.9 LED Interface

The 82598 provides four LEDs per port that can be used to indicate the status of the traffic. The following parameters can be defined for each of the LEDs:

1. Mode: defines which information is reflected by this LED. The encoding is described in the LEDCTL register.
2. Polarity: defines the polarity of the LED.
3. Blink mode: should the LED 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 LEDs.

Note: See Section 3.5.7 for a more detailed description of LED behavior.

§ §

2. Signal Descriptions and Pinout List

Signal names are subject to change without notice. Verify with your local Intel sales office that you have the latest information before finalizing a design.

2.1 Signal Type Definitions

Signals are electrically defined in Table 2-1.

Table 2-1. Signal Definitions

Name	Definition
I	Input Standard input only digital signal.
Out (O)	Output Totem Pole Output (TPO) is a standard active driver.
T/s	Tri-state Bi-directional three-state digital input/output pin.
O/d	Open Drain Enables multiple devices to share as a wire-OR.
A-in	Analog input signals.
A-out	Analog output signals.
A-Inout	Bi-directional analog signals.
B	Input BIAS.
NCSI-in	NC-SI input signal.
NCSI-out	NC-SI output signal.
Pu	Internal pull-up
Pd	Internal pull-down

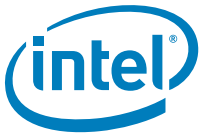
Table 2-2. Reserved and No-Connect Definitions

Name	Definition
No Connect (NC)	These package balls are not connected.
Reserved No Connect (RSVD_NC)	These package balls are connected, but are reserved for internal use. They should be left floating on the board-level design.
Reserved 1P2 (RSVD_1P2)	These package balls are connected, but are reserved for internal use. They should be connected to 1.2 V_LAN on the board-level design.
Reserved VSS (RSVD_VSS)	These package balls are connected, but are reserved for internal use. They should be connected to GND on the board-level design.

2.2 PCIe Interface

Table 2-3. PCIe Signal and Pin Information

Signal	Pin Number	Type	Name and Function
PE_CLKP PE_CLKN	AJ28 AK28	A-in	PCIe Differential Reference Clock In. A 100 MHz differential clock input. This clock is used as the reference clock for the PCIe Tx/Rx circuitry and by the PCIe core PLL to generate clocks for the PCIe core logic.
PET_0_P PET_0_N	AH29 AH30	A-out	PCIe Serial Data Output. A serial differential output pair 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.
PET_1_P PET_1_N	AE29 AE30	A-out	PCIe Serial Data Output. A serial differential output pair 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.
PET_2_P PET_2_N	AB29 AB30	A-out	PCIe Serial Data Output. A serial differential output pair 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.
PET_3_P PET_3_N	W29 W30	A-out	PCIe Serial Data Output. A serial differential output pair 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.
PET_4_P PET_4_N	N29 N30	A-out	PCIe Serial Data Output. A serial differential output pair 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.
PET_5_P PET_5_N	K29 K30	A-out	PCIe Serial Data Output. A serial differential output pair 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.
PET_6_P PET_6_N	G29 G30	A-out	PCIe Serial Data Output. A serial differential output pair 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.
PET_7_P PET_7_N	D29 D30	A-out	PCIe Serial Data Output. A serial differential output pair 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.



Signal	Pin Number	Type	Name and Function
PER_0_P PER_0_N	AG29 AG30	A-in	PCIe Serial Data Input. A serial differential input pair running at 2.5 Gb/s. An embedded clock present in this input is recovered along with the data.
PER_1_P PER_1_N	AD29 AD30	A-in	PCIe Serial Data Input. A serial differential input pair running at 2.5Gb/s. An embedded clock present in this input is recovered along with the data.
PER_2_P PER_2_N	AA29 AA30	A-in	PCIe Serial Data Input. A serial differential input pair running at 2.5Gb/s. An embedded clock present in this input is recovered along with the data.
PER_3_P PER_3_N	V29 V30	A-in	PCIe Serial Data Input. A serial differential input pair running at 2.5Gb/s. An embedded clock present in this input is recovered along with the data.
PER_4_P PER_4_N	M29 M30	A-in	PCIe Serial Data Input. A serial differential input pair running at 2.5Gb/s. An embedded clock present in this input is recovered along with the data.
PER_5_P PER_5_N	J29 J30	A-in	PCIe Serial Data Input. A serial differential input pair running at 2.5Gb/s. An embedded clock present in this input is recovered along with the data.
PER_6_P PER_6_N	F29 F30	A-in	PCIe Serial Data Input. A serial differential input pair running at 2.5Gb/s. An embedded clock present in this input is recovered along with the data.
PER_7_P PER_7_N	C29 C30	A-in	PCIe Serial Data Input. A serial differential input pair running at 2.5Gb/s. An embedded clock present in this input is recovered along with the data.
PE_RCOMP_N PE_RCOMP_P	R29 R28	B	Impedance Compensation. Should be connected with an external 1.4 KΩ ±1%, 100 ppm resistor.
PE_RST_N	AK27	I	Power and Clock Good Indication. Indicates that power and the PCIe reference clock are within specified values. Defined in the PCIe specifications.
PE_WAKE_N	AG28	O/d	Wake. Pulled to 0b to indicate that a Power Management Event (PME) is pending and the PCIe link should be restored. Defined in the PCIe specifications.

2.3 XAUI Interface Signals

Table 2-4. Signal and Pin Information

Signal	Pin Number	Type	Name and Function
RBIAS RSENSE	AG2 AF1	B	• RBIAS Resistor. A 6.5 KΩ resistor must be connected between RBIAS and GND for proper operation. This resistor generates internal bias currents. • RSENSE is an internal sense point and must be connected to the ground connection of the 6.5 KΩ RBias resistor, as close to the package as possible.
REFCLKIN_P REFCLKIN_N	AK3 AJ3	A-in	External Reference Clock Input. Must be connected to a 156.25 MHz +/-0.005% (+/- 50 ppm) clock source. If an external clock is to be applied, it must be 156.25 MHz +/-0.005% (+/- 50 ppm). Adequate board layout is required to avoid clock waveform reflections and glitches.

Signal	Pin Number	Type	Name and Function
RX0_L3_P RX0_L3_N	AJ23 AK23	A-in	XAUI serial data input for port 0. A serial differential input pair running at up to 3.125 Gb/s. An embedded clock present in this input is recovered along with the data.
RX0_L2_P RX0_L2_N	AJ24 AK24	A-in	XAUI serial data input for port 0. A serial differential input pair running at up to 3.125 Gb/s. An embedded clock present in this input is recovered along with the data.
RX0_L1_P RX0_L1_N	AJ25 AK25	A-in	XAUI serial data input for port 0. A serial differential input pair running at up to 3.125 Gb/s. An embedded clock present in this input is recovered along with the data.
RX0_L0_P RX0_L0_N	AJ26 AK26	A-in	XAUI serial data input for port 0. A serial differential input pair running at up to 3.125 Gb/s. An embedded clock present in this input is recovered along with the data.
TX0_L3_P TX0_L3_N	AJ18 AK18	A-out	XAUI serial data output for port 0. A serial differential output pair running at up to 3.125 Gb/s. This output carries both data and an embedded clock that is recovered along with data at the receiving end.
TX0_L2_P TX0_L2_N	AJ19 AK19	A-out	XAUI serial data output for port 0. A serial differential output pair running at up to 3.125 Gb/s. This output carries both data and an embedded clock that is recovered along with data at the receiving end.
TX0_L1_P TX0_L1_N	AJ20 AK20	A-out	XAUI serial data output for port 0. A serial differential output pair running at up to 3.125 Gb/s. This output carries both data and an embedded clock that is recovered along with data at the receiving end.
TX0_L0_P TX0_L0_N	AJ21 AK21	A-out	XAUI serial data output for port 0. A serial differential output pair running at up to 3.125 Gb/s. This output carries both data and an embedded clock that is recovered along with data at the receiving end.
RX1_L3_P RX1_L3_N	AJ10 AK10	A-in	XAUI serial data input for port 1. A serial differential input pair running at up to 3.125 Gb/s. An embedded clock present in this input is recovered along with the data.
RX1_L2_P RX1_L2_N	AJ11 AK11	A-in	XAUI serial data input for port 1. A serial differential input pair running at up to 3.125 Gb/s. An embedded clock present in this input is recovered along with the data.
RX1_L1_P RX1_L1_N	AJ12 AK12	A-in	XAUI serial data input for port 1. A serial differential input pair running at up to 3.125 Gb/s. An embedded clock present in this input is recovered along with the data.
RX1_L0_P RX1_L0_N	AJ13 AK13	A-in	XAUI serial data input for port 1. A serial differential input pair running at up to 3.125 Gb/s. An embedded clock present in this input is recovered along with the data.
TX1_L3_P TX1_L3_N	AJ5 AK5	A-out	XAUI serial data output for port 1. A serial differential output pair running at up to 3.125 Gb/s. This output carries both data and an embedded clock that is recovered along with data at the receiving end.
TX1_L2_P TX1_L2_N	AJ6 AK6	A-out	XAUI serial data output for port 1. A serial differential output pair running at up to 3.125 Gb/s. This output carries both data and an embedded clock that is recovered along with data at the receiving end.
TX1_L1_P TX1_L1_N	AJ7 AK7	A-out	XAUI serial data output for port 1. A serial differential output pair running at up to 3.125 Gb/s. This output carries both data and an embedded clock that is recovered along with data at the receiving end.
TX1_L0_P TX1_L0_N	AJ8 AK8	A-out	XAUI serial data output for port 1. A serial differential output pair running at up to 3.125 Gb/s. This output carries both data and an embedded clock that is recovered along with data at the receiving end.

2.4 EEPROM and Serial Flash Interface Signals

Table 2-5. EEPROM Signals

Signal	Pin Number	Type	Name and Function
EE_DI	A5	T/s	Data output to EEPROM.
EE_DO	B6	In	Data input from EEPROM.
EE_SK	A6	T/s	EEPROM serial clock that operates at a maximum of 2 MHz.
EE_CS_N	B7	T/s	EEPROM chip select output.

Table 2-6. Serial Flash Signals

Signal	Pin Number	Type	Name and Function
FLSH_SI	A8	T/s	Serial data output to the Flash.
FLSH_SO	A7	In	Serial data input from the Flash.
FLSH_SCK	B9	T/s	Flash serial clock that operates at a maximum of 20 MHz.
FLSH_CE_N	B8	T/s	Flash chip select output.

2.5 SMBus and NC-SI Signals

Table 2-7. SMBus Signals

Signal	Pin Number	Type	Name and Function
SMBCLK	AJ27	O/d	SMBus Clock. One clock pulse is generated for each data bit transferred.
SMBD	AH28	O/d	SMBus Data. Stable during the high period of the clock (unless it is a start or stop condition).
SMBALRT_N	AE3	O/d	SMBus Alert. Acts as an interrupt pin of a slave device on the SMBus.

Note: If the SMBus is disconnected, an external pull-up should be used for SMBCLK and SMBD pins. For suggested pull-up resistor values, refer to Section 2.13.

Table 2-8. NC-SI Signals

Symbol	Pin Number	Type	Name and Function
NCSI_CLK_IN	B20	NCSI-In	NC-SI Reference Clock Input. Synchronous clock reference for receive, transmit, and control interface. It is a 50 MHz clock/- 50 ppm.

NCSI_CRS_DV	A19	NCSI-Out	CRS/DV. Carrier sense/receive data valid.
NCSI_RXD_0 NCSI_RXD_1	B18 B19	NCSI-Out	Receive Data. Data signals to the BMC.
NCSI_TX_EN	A17	NCSI-In	Transmit Enable.
NCSI_TXD_0 NCSI_TXD_1	A20 A18	NCSI-In	Transmit Data. Data signals from the BMC.

Note: If NC-SI is disconnected, an external pull-up resistor should be connected to the NCSI_TXD[1:0] and an external pull down resistor should be connected to the NCSI_CLK_IN and NCSI_TX_EN pins. For suggested pull-up/pull-down values, refer to Section 2.13.

For more information on management interfaces, refer Section 5..

2.6 MDI/O Signals

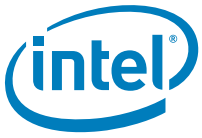
Table 2-9. MDI/O

Symbol	Pin Number	Type	Name and Function
MDIO0	AE2	O/d	Mgmt Data. Bi-directional signal for serial data transfers between the 82598 and the PHY management registers for port 0. **Note:** Requires an external pull-up device.
MDC0	AD1	O	Mgmt Clock. Clock output for accessing the PHY management registers for port 0. Nominal frequency can be set to 2.4 MHz (default) or 24 MHz.
MDIO1	AD2	O/d	Mgmt Data. Bi-directional signal for serial data transfers between the 82598 and the PHY management registers for port 1. **Note:** Requires an external pull-up device.
MDC1	AE1	O	Mgmt Clock. Clock output for accessing the PHY management registers for port 1. Nominal frequency can be set to 2.4 MHz (default) or 24 MHz.

2.7 Software-Definable Pins

Table 2-10. Software-Defined Pins

Symbol	Pin Number	Type	Name and Function
SDP0_0 SDP0_1 SDP0_2 SDP0_3 SDP0_4 SDP0_5	W2 V2 V3 U2 U3 T2	T/s	General purpose software-defined pins for function 0.
SDP0_6 SDP0_7	T3 R2	O/d	General purpose O/D software-defined pins for function 0.



SDP1_0 SDP1_1 SDP1_2 SDP1_3 SDP1_4 SDP1_5	R3 P2 P3 N2 M1 M2	T/s	General purpose software-defined pins for function 1.
SDP1_6 SDP1_7	L1 L2	O/d	General purpose O/D software-defined pins for function 1.

2.8 LED Signals

Table 2-11. LED Signals

Symbol	Pin Number	Type	Name and Function
LED0_0	AC1	O	Port 0 LED0. By default, programmable LED that indicates link-up.
LED0_1	AC2	O	Port 0 LED1. Programmable LED that indicates 10 Gb/s link.
LED0_2	AB2	O	Port 0 LED2. By default, programmable LED that indicates a link/activity indication.
LED0_3	AA1	O	Port 0 LED3. By default, programmable LED that indicates a 1 Gb/s link.
LED1_0	AA2	O	Port 1 LED0. By default, programmable LED that indicates link-up.
LED1_1	Y1	O	Port 1 LED1. By default, programmable LED that indicates 10 Gb/s link.
LED1_2	Y2	O	Port 1 LED2. By default, programmable LED that indicates a link/activity indication.
LED1_3	W1	O	Port 1 LED3. By default, programmable LED that indicates a 1 Gb/s link.

2.9 Miscellaneous Signals

Table 2-12. Miscellaneous Signals

Symbol	Pin Number	Type	Name and Function
LAN1_DIS_N	A11	T/s	This pin is a strapping pin latched at the rising edge of LAN_PWR_GOOD, Internal Power On Reset, PE_RST_N, or in-band PCIe reset. If this pin is not connected or driven high during initialization, LAN 1 is enabled. If this pin is driven low during initialization, LAN 1 port is disabled.
PHY0_PWRDN_N	B15	O	This pin controls the ability to put external PHY 0 in power down mode according to the 82598's internal power state.
PHY1_PWRDN_N	A12	O	This pin controls the ability to put external PHY 1 in power down mode according to the 82598's internal power state.

POR_BYPASS	AC3	In	Bypass indication as to whether or not to use Internal Power On Reset or the LAN_PWR_GOOD pin. When high, the 82598 disables the Internal Power On Reset circuit and uses the LAN_PWR_GOOD pin as the power on reset indication.
MAIN_PWR_OK	B12	In	Main Power OK**.** Indicates that platform main power is up. Must be connected externally.
DEV_PWRDN_N	B13	O	This pin can control the external power supply to the 82598 according to the internal power state using an external circuitry.
LAN0_DIS_N	B14	T/s	This pin is a strapping pin latched at the rising edge of LAN_PWR_GOOD, Internal Power On Reset, PE_RST_N, or in-band PCIe reset. If this pin is not connected or driven high during initialization, LAN 0 is enabled. If this pin is driven low during initialization, LAN 0 port is disabled.
AUX_PWR	B16	T/s	Auxiliary Power Available. When set, indicates that auxiliary power is available and the 82598 should support $D3_{COLD}$ power state if enabled to do so. This pin is latched at the rising edge of Internal Power On Reset or LAN_PWR_GOOD.
LAN_PWR_GOOD	B17	In	LAN_PWR_GOOD. A transition from low to high initializes the 82598 by resetting it. This pin is used in conjunction with POR_BYPASS. For the pin to operate correctly, the LAN_PWR_GOOD circuit needs to be bypassed (POR_BYPASS = 1b).

2.10 Test Interface Signals

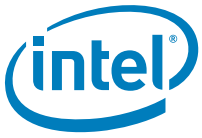
Table 2-13. Test Interface Signals

Symbol	Pin Number	Type	Name and Function
JTCK	F2	In	JTAG Clock Input.
JTDI	D2	In	JTAG Data Input.
JTDO	F1	O/d	JTAG Data Output.
JTMS	E2	In	JTAG TMS Input.
JRST_N	C2	In	JTAG Reset Input. Active low reset for the JTAG port.

2.11 Power Supplies

Table 2-14. Digital and Analog Supplies

Symbol	Pin Number	Type	Name and Function
VCC3P3	AB1, V1, N1, H1, C1.	3.3 V dc	3.3 V dc Power Input.



VCC1P2	Y20, Y19, Y13, Y12, Y9, Y8, Y7, Y6, V20, V19, V18, V17, V16, V15, V14, V13, V12, V10, V9, V8, V7, V6, T20, T19, T18, T16, T15, T14, T13, T12, T11, T10, T9, T8, T7, T6, P20, P19, P18, P17, P16, P15, P14, P13, P12, P11, P10, P9, P8, P7, P6, M20, M19, M18, M17, M16, M15, M14, M13, M12, M11, M10, M9, M8, M7, M6, K20, K19, K18, K17, K16, K15, K14, K13, K12, K10, K9, K8, K7, K6, H20, H19, H18, H17, H16, H15, H14, H13, H12, H11, H10, H9, H8, H7, H6, F19, F18, F17, F16, F15, F14, F13, F12, F11, F10, F9, F8, F7, F6, D19, D18, D17, D16, D15, D14, D13, D12, D11, D10, D9, D8, D7, D6.	1.2 V dc	1.2 V dc Power Input.
VSS	AE5, AE4, AD6, AD5, AD4, AC8, AC7, AC6, AC5, AC4, AB11, AB10, AB9, AB8, AB7, AB6, AB5, AB4, AA12, AA11, AA10, AA9, AA8, AA7, AA6, AA5, AA4, Y5, Y4, W20, W19, W18, W17, W16, W15, W14, W13, W12, W10, W9, W8, W7, W6, W5, W4, V5, U20, U19, U18, U17, U16, U15, U14, U13, U12, U11, U10, U9, U8, U7, U6, U5, T5, R20, R19, R18, R16, R15, R14, R13, R12, R11, R10, R9, R8, R7, R6, R5, P5, N20, N19, N18, N17, N16, N15, N14, N13. N12, N11, N10, N9, N8, N7, N6, N5, M5, L20, L19, L18, L17, L16, L15, L14, L13, L12, L10, L9, L8, L7, L6, L5, K5, J18, J17, J16, J15, J14, J13, J12, J11, J10, J9, J8, J7, J6, J5, H5, G20, G19, G18, G17, G16, G15, G14, G13, G12, G11, G10, G9, G8, G7, G6, G5, F5, E20, E19, E18, E17, E16, E15, E14, E13, E12, E11, E10, E9, E8, E7, E6, E5, D5, C20, C19, C18, C17, C16, C15, C14, C13, C12, C11, C10, C9, C8, C7, C6, C5, B2, B1, A2.	0 V dc	Core Ground (Core and PE-VSS).
VCC1P2	AJ16, AJ15, AH16, AH15, AG24, AG23, AG22, AG21, AG20, AG19, AG18, AG17, AG16, AG15, AG14, AG13, AG12, AG11, AG10, AG9, AG8, AG7, AG6, AF16, AF15, AE22, AE21, AE20, AE19, AE18, AE17, AE16, AE15, AE14, AE13, AE12, AE11, AE10, AE9, AD16, AD15, AC20, AC19, AC18, AC17, AC16, AC15, AA19, AA18, AA17, AA16, AA15, Y18, Y17, Y16, Y15.	1.2 V dc	XAUI 1.2 V dc Analog Power Supply.
VCC1P2	AH2, AH1, AG5, AG4, AG3, AF6, AE8, AE7, AD11, AD10, AD9, AC12, AB13, AA14.	1.2 V dc	XAUI Common 1.2 V dc Power Supply.
VSS	AK30, AK29, AK22, AK9, AK4, AJ30, AJ29, AJ22, AJ17, AJ14, AJ9, AJ4, AH27, AH26, AH25, AH24, AH23, AH22, AH21, AH20, AH19, AH18, AH17, AH14, AH13, AH12, AH11, AH10, AH9, AH8, AH7, AH6, AH5, AG26, AG25, AF25, AF24, AF23, AF22, AF21, AF20, AF19, AF18, AF17, AF14, AF13, AF12, AF11, AF10, AF9, AF8, AF7, AE24, AE23, AD23, AD22, AD21, AD20, AD19, AD18, AD17, AD14, AD13, AD12, AC22, AC21, AB21, AB20, AB19, AB18, AB17, AB16, AB15, AB14.	0 V dc	XAUI Digital/ Analog Ground.
VSS	AK2, AK1, AJ2, AJ1, AH4, AH3, AF5, AF4, AF3, AF2, AE6, AD8, AD7, AC11, AC10, AC9, AB12, AA13, Y14.	0 V dc	XAUI Common Ground.
VCC1P2	AD27, AC27, AC25, AB27, AB25, AA27, AA25, AA23, Y27, Y25, Y23, Y21, W27, W25, W23, W21, V27, V25, V23, U27, U25, U23, T27, T25, T23, R27, R25, R23, P27, P25, N27, N25, M27, L27.	1.2 V dc	1.2 V dc for PCIe* Circuits.
VCC1P8	V21, U21, T21, R21, P23, P21, N23, N21, M25, M23, M21, L25, L23, L21, K27, K25, K23, K21, J27, J25, J23, J21, H27, H25, H23, H21, G27, G25, G23, G21, F27, F25, F23, F21, E27, E25, E23, E21, D27, D25, D23, D21, C27, C25, C23, C21, B27, B25, B23, B21, A27, A25, A23, A21.	1.8 V dc	1.8 V dc for PCIe* Circuits.
Symbol	**Pin Number**	**Type**	**Name and Function**
VSS	AG27, AF30, AF29, AF28, AF27, AF26, AE28, AE27, AE26, AD28, AD26, AD25, AC30, AC29, AC28, AC26, AC24, AC23, AB28, AB26, AB24, AB23, AB22, AA28, AA26, AA24, AA22, AA21, AA20, Y30, Y29, Y28, Y26, Y24, Y22, W28, W26, W24, W22, V26, V24, V22, U26, U24, U22, T26, T24, T22, R26, R24, R22, P26, P24, P22, N26, N24, N22, M28, M26, M24, M22, L30, L29, L28, L26, L24, L22, K28, K26, K24, K22, J28, J26, J24, J22, H30, H29, H28, H26, H24, H22, G28, G26, G24, G22, F28, F26, F24, F22, E30, E29, E28, E26, E24, E22, D28, D26, D24, D22, C28, C26, C24, C22, B30, B29, B28, B26, B24, B22, A30, A29, A28, A26, A24, A22.	0 V dc	PE Ground.

2.12 Alphabetical Pinout/Signal Name

Table 2-15 lists the signal name associated with each pin.

Note: The signal names are subject to change without notice. Verify with your local Intel sales office that you have the latest information before finalizing a design.

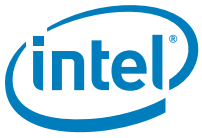
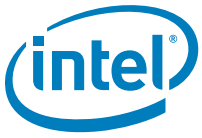


Table 2-15. Alphabetical Pin Name/Signal Name

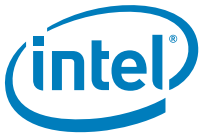
Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
A2	VSS	B12	MAIN_PWR_OK	C22	VSS
A3	RSVDA3_NC	B13	DEV_PWRDN_N	C23	VCC1P8
A4	RSVDA4_NC	B14	LAN0_DIS_N	C24	VSS
A5	EE_DI	B15	PHY0_PWRDN_N	C25	VCC1P8
A6	EE_SK	B16	AUX_PWR	C26	VSS
A7	FLSH_SO	B17	LAN_PWR_GOOD	C27	VCC1P8
A8	FLSH_SI	B18	NCSI_RXD_0	C28	VSS
A9	RSVDA9_NC	B19	NCSI_RXD_1	C29	PER_7_P
A10	RSVDA10_NC	B20	NCSI_CLK_IN	C30	PER_7_N
A11	LAN1_DIS_N	B21	VCC1P8	D1	RSVDD1_NC
A12	PHY1_PWRDN_N	B22	VSS	D2	JTDI
A13	(Blank)	B23	VCC1P8	D3	RSVDD3_NC
A14	(Blank)	B24	VSS	D4	RSVDD4_NC
A15	(Blank)	B25	VCC1P8	D5	VSS
A16	(Blank)	B26	VSS	D6	VCC1P2
A17	NCSI_TX_EN	B27	VCC1P8	D7	VCC1P2
A18	NCSI_TXD_1	B28	VSS	D8	VCC1P2
A19	NCSI_CRS_DV	B29	VSS	D9	VCC1P2
A20	NCSI_TXD_0	B30	VSS	D10	VCC1P2
A21	VCC1P8	C1	VCC3P3	D11	VCC1P2
A22	VSS	C2	JRST_N	D12	VCC1P2
A23	VCC1P8	C3	RSVDC3_NC	D13	VCC1P2
A24	VSS	C4	RSVDC4_NC	D14	VCC1P2
A25	VCC1P8	C5	VSS	D15	VCC1P2
A26	VSS	C6	VSS	D16	VCC1P2
A27	VCC1P8	C7	VSS	D17	VCC1P2
A28	VSS	C8	VSS	D18	VCC1P2
A29	VSS	C9	VSS	D19	VCC1P2
A30	VSS	C10	VSS	D20	RSVDD20_NC

Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
B1	VSS	C11	VSS	D21	VCC1P8
B2	VSS	C12	VSS	D22	VSS
B3	RSVDB3_NC	C13	VSS	D23	VCC1P8
B4	RSVDB4_NC	C14	VSS	D24	VSS
B5	RSVDB5_NC	C15	VSS	D25	VCC1P8
B6	EE_DO	C16	VSS	D26	VSS
B7	EE_CS_N	C17	VSS	D27	VCC1P8
B8	FLSH_CE_N	C18	VSS	D28	VSS
B9	FLSH_SCK	C19	VSS	D29	PET_7_P
B10	RSVDB10_NC	C20	VSS	D30	PET_7_N
B11	RSVDB11_NC	C21	VCC1P8	E1	RSVDE1_VSS
E2	JTMS	F14	VCC1P2	G24	VSS
E3	RSVDE3_NC	F15	VCC1P2	G25	VCC1P8
E4	RSVDE4_NC	F16	VCC1P2	G26	VSS
E5	VSS	F17	VCC1P2	G27	VCC1P8
E6	VSS	F18	VCC1P2	G28	VSS
E7	VSS	F19	VCC1P2	G29	PET_6_P
E8	VSS	F20	RSVDF20_NC	G30	PET_6_N
E9	VSS	F21	VCC1P8	H1	VCC3P3
E10	VSS	F22	VSS	H2	RSVDH2_NC
E11	VSS	F23	VCC1P8	H3	RSVDH3_NC
E12	VSS	F24	VSS	H4	RSVDH4_NC
E13	VSS	F25	VCC1P8	H5	VSS
E14	VSS	F26	VSS	H6	VCC1P2
E15	VSS	F27	VCC1P8	H7	VCC1P2
E16	VSS	F28	VSS	H8	VCC1P2
E17	VSS	F29	PER_6_P	H9	VCC1P2
E18	VSS	F30	PER_6_N	H10	VCC1P2
E19	VSS	G1	RSVDG1_NC	H11	VCC1P2
E20	VSS	G2	RSVDG2_NC	H12	VCC1P2



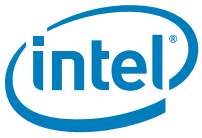
Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
E21	VCC1P8	G3	RSVDG3_NC	H13	VCC1P2
E22	VSS	G4	RSVDG4_NC	H14	VCC1P2
E23	VCC1P8	G5	VSS	H15	VCC1P2
E24	VSS	G6	VSS	H16	VCC1P2
E25	VCC1P8	G7	VSS	H17	VCC1P2
E26	VSS	G8	VSS	H18	VCC1P2
E27	VCC1P8	G9	VSS	H19	VCC1P2
E28	VSS	G10	VSS	H20	VCC1P2
E29	VSS	G11	VSS	H21	VCC1P8
E30	VSS	G12	VSS	H22	VSS
F1	JTDO	G13	VSS	H23	VCC1P8
F2	JTCK	G14	VSS	H24	VSS
F3	RSVDF3_NC				
F4	RSVDF4_NC	G15	VSS	H25	VCC1P8
F5	VSS	G16	VSS	H26	VSS
F6	VCC1P2	G17	VSS	H27	VCC1P8
F7	VCC1P2	G18	VSS	H28	VSS
F8	VCC1P2	G19	VSS	H29	VSS
F9	VCC1P2	G2	RSVDG2_NC	H30	VSS
F10	VCC1P2	G20	VSS	J1	RSVDJ1_VSS
F11	VCC1P2	G21	VCC1P8	J2	RSVDJ2_NC
F12	VCC1P2	G22	VSS	J3	RSVDJ3_NC
F13	VCC1P2	G23	VCC1P8	J4	RSVDJ4_NC
J5	VSS	K16	VCC1P2	L28	VSS
J6	VSS	K17	VCC1P2	L29	VSS
J7	VSS	K18	VCC1P2	L30	VSS
J8	VSS	K19	VCC1P2	M1	SDP1_4
J9	VSS	K20	VCC1P2	M2	SDP1_5
J10	VSS	K21	VCC1P8	M3	NCM3
J11	VSS	K22	VSS	M4	RSVDM4_NC

Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
J12	VSS	K23	VCC1P8	M5	VSS
J13	VSS	K24	VSS	M6	VCC1P2
J14	VSS	K25	VCC1P8	M7	VCC1P2
J15	VSS	K26	VSS	M8	VCC1P2
J16	VSS	K27	VCC1P8	M9	VCC1P2
J17	VSS	K28	VSS	M10	VCC1P2
J18	VSS	K29	PET_5_P	M11	VCC1P2
J19	NCJ19	K30	PET_5_N	M12	VCC1P2
J20	NCJ20	L1	SDP1_6	M13	VCC1P2
J21	VCC1P8	L2	SDP1_7	M14	VCC1P2
J22	VSS	L3	NCL3	M15	VCC1P2
J23	VCC1P8	L4	RSVDL4_NC	M16	VCC1P2
J24	VSS	L5	VSS	M17	VCC1P2
J25	VCC1P8	L6	VSS	M18	VCC1P2
J26	VSS	L7	VSS	M19	VCC1P2
J27	VCC1P8	L8	VSS	M20	VCC1P2
J28	VSS	L9	VSS	M21	VCC1P8
		L10	VSS		
J29	PER_5_P	L11	NCL11	M22	VSS
J30	PER_5_N	L12	VSS	M23	VCC1P8
K1	NCK1	L13	VSS	M24	VSS
K2	RSVDK2_NC	L14	VSS	M25	VCC1P8
K3	RSVDK3_NC	L15	VSS	M26	VSS
K4	RSVDK4_NC	L16	VSS	M27	VCC1P2
K5	VSS	L17	VSS	M28	VSS
K6	VCC1P2	L18	VSS	M29	PER_4_P
K7	VCC1P2	L19	VSS	M30	PER_4_N
K8	VCC1P2	L20	VSS	N1	VCC3P3
K9	VCC1P2	L21	VCC1P8	N2	SDP1_3
K10	VCC1P2	L22	VSS	N3	NCN3



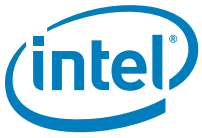
Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
K11	NCK11	L23	VCC1P8	N4	RSVDN4_NC
K12	VCC1P2	L24	VSS	N5	VSS
K13	VCC1P2	L25	VCC1P8	N6	VSS
K14	VCC1P2	L26	VSS	N7	VSS
K15	VCC1P2	L27	VCC1P2	N8	VSS
				N9	VSS
N10	VSS	P21	VCC1P8	T1	(blank)
N11	VSS	P22	VSS	T2	SDP0_5
N12	VSS	P23	VCC1P8	T3	SDP0_6
N13	VSS	P24	VSS	T4	NCT4
N14	VSS	P25	VCC1P2	T5	VSS
N15	VSS	P26	VSS	T6	VCC1P2
N16	VSS	P27	VCC1P2	T7	VCC1P2
N17	VSS	P28	NCP28	T8	VCC1P2
N18	VSS	P29	NCP29	T9	VCC1P2
N19	VSS	P30	(blank)	T10	VCC1P2
N20	VSS	R1	(blank)	T11	VCC1P2
N21	VCC1P8	R2	SDP0_7	T12	VCC1P2
N22	VSS	R3	SDP1_0	T13	VCC1P2
N23	VCC1P8	R4	NCR4	T14	VCC1P2
N24	VSS	R5	VSS	T15	VCC1P2
N25	VCC1P2	R6	VSS	T16	VCC1P2
N26	VSS	R7	VSS	T17	NCT17
N27	VCC1P2	R8	VSS	T18	VCC1P2
N28	NCN28	R9	VSS	T19	VCC1P2
N29	PET_4_P	R10	VSS	T20	VCC1P2
N30	PET_4_N	R11	VSS	T21	VCC1P8
P1	(blank)	R12	VSS	T22	VSS
P2	SDP1_1	R13	VSS	T23	VCC1P2
P3	SDP1_2	R14	VSS	T24	VSS

Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
P4	NCP4	R15	VSS	T25	VCC1P2
P5	VSS	R16	VSS	T26	VSS
P6	VCC1P2	R17	NCR17	T27	VCC1P2
P7	VCC1P2	R18	VSS	T28	RSVDT28_NC
P8	VCC1P2	R19	VSS	T29	RSVDT29_NC
P9	VCC1P2	R2	SDP0_7	T30	(blank)
P10	VCC1P2	R20	VSS	U1	(blank)
P11	VCC1P2	R21	VCC1P8	U2	SDP0_3
P12	VCC1P2	R22	VSS	U3	SDP0_4
P13	VCC1P2	R23	VCC1P2	U4	RSVDU4_NC
P14	VCC1P2	R24	VSS	U5	VSS
P15	VCC1P2	R25	VCC1P2	U6	VSS
P16	VCC1P2	R26	VSS	U7	VSS
P17	VCC1P2	R27	VCC1P2	U8	VSS
P18	VCC1P2	R28	PE_RCOMP_P	U9	VSS
P19	VCC1P2	R29	PE_RCOMP_N	U10	VSS
P20	VCC1P2	R30	(blank)	U11	VSS
U12	VSS	V23	VCC1P2	Y4	VSS
U13	VSS	V24	VSS	Y5	VSS
U14	VSS	V25	VCC1P2	Y6	VCC1P2
U15	VSS	V26	VSS	Y7	VCC1P2
U16	VSS	V27	VCC1P2	Y8	VCC1P2
U17	VSS	V28	NCV28	Y9	VCC1P2
U18	VSS	V29	PER_3_P	Y10	RSVDY10_NC
U19	VSS	V30	PER_3_N	Y11	RSVDY11_NC
U20	VSS	W1	LED1_3	Y12	VCC1P2
U21	VCC1P8	W2	SDP0_0	Y13	VCC1P2
U22	VSS	W3	RSVDW3_NC	Y14	VSS
U23	VCC1P2	W4	VSS	Y15	VCC1P2
U24	VSS	W5	VSS	Y16	VCC1P2



Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
U25	VCC1P2	W6	VSS	Y17	VCC1P2
U26	VSS	W7	VSS	Y18	VCC1P2
U27	VCC1P2	W8	VSS	Y19	VCC1P2
U28	NCU28	W9	VSS	Y20	VCC1P2
U29	NCU29	W10	VSS	Y21	VCC1P2
U30	(blank)	W11	NCW11	Y22	VSS
V1	VCC3P3	W12	VSS	Y23	VCC1P2
V2	SDP0_1	W13	VSS	Y24	VSS
V3	SDP0_2	W14	VSS	Y25	VCC1P2
V4	RSVDV4_NC	W15	VSS	Y26	VSS
V5	VSS	W16	VSS	Y27	VCC1P2
V6	VCC1P2	W17	VSS	Y28	VSS
V7	VCC1P2	W18	VSS	Y29	VSS
V8	VCC1P2	W19	VSS	Y30	VSS
V9	VCC1P2	W20	VSS	AA1	LED0_3
V10	VCC1P2	W21	VCC1P2	AA2	LED1_0
V11	NCV11	W22	VSS	AA3	RSVDAA3_NC
V12	VCC1P2	W23	VCC1P2	AA5	VSS
V13	VCC1P2	W24	VSS	AA6	VSS
V14	VCC1P2	W25	VCC1P2	AA7	VSS
V15	VCC1P2	W26	VSS	AA8	VSS
V16	VCC1P2	W27	VCC1P2	AA9	VSS
V17	VCC1P2	W28	VSS	AA10	VSS
V18	VCC1P2	W29	PET_3_P	AA11	VSS
V19	VCC1P2	W30	PET_3_N	AA12	VSS
V20	VCC1P2	Y1	LED1_1	AA13	VSS
V21	VCC1P8	Y2	LED1_2	AA14	VCC1P2
V22	VSS	Y3	RSVDY3_NC	AA15	VCC1P2
AA16	VCC1P2	AB27	VCC1P2	AD8	VSS
AA17	VCC1P2	AB28	VSS	AD9	VCC1P2

Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
AA18	VCC1P2	AB29	PET_2_P	AD10	VCC1P2
AA19	VCC1P2	AB30	PET_2_N	AD11	VCC1P2
AA20	VSS	AC1	LED0_0	AD12	VSS
AA21	VSS	AC2	LED0_1	AD13	VSS
AA22	VSS	AC3	POR_BYPASS	AD14	VSS
AA23	VCC1P2	AC4	VSS	AD15	VCC1P2
AA24	VSS	AC5	VSS	AD16	VCC1P2
AA25	VCC1P2	AC6	VSS	AD17	VSS
AA26	VSS	AC7	VSS	AD18	VSS
AA27	VCC1P2	AC8	VSS	AD19	VSS
AA28	VSS	AC9	VSS	AD20	VSS
AA29	PER_2_P	AC10	VSS	AD21	VSS
AA30	PER_2_N	AC11	VSS	AD22	VSS
AB1	VCC3P3	AC12	VCC1P2	AD23	VSS
AB2	LED0_2	AC13	RSVDAC13_NC	AD24	RSVDAD24_NC
AB3	RSVDAB3_NC	AC14	RSVDAC14_NC	AD25	VSS
AB4	VSS	AC15	VCC1P2	AD26	VSS
AB5	VSS	AC16	VCC1P2	AD27	VCC1P2
AB6	VSS	AC17	VCC1P2	AD28	VSS
AB7	VSS	AC18	VCC1P2	AD29	PER_1_P
AB8	VSS	AC19	VCC1P2	AD30	PER_1_N
AB9	VSS	AC20	VCC1P2	AE1	MDC1
AB10	VSS	AC21	VSS	AE2	MDIO0
AB11	VSS	AC22	VSS	AE3	SMBALRT_N
AB12	VSS	AC23	VSS	AE4	VSS
AB13	VCC1P2	AC24	VSS	AE5	VSS
AB14	VSS	AC25	VCC1P2	AE6	VSS
AB15	VSS	AC26	VSS	AE7	VCC1P2
AB16	VSS	AC27	VCC1P2	AE8	VCC1P2
AB17	VSS	AC28	VSS	AE9	VCC1P2



Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
AB18	VSS	AC29	VSS	AE10	VCC1P2
AB19	VSS	AC30	VSS	AE11	VCC1P2
AB20	VSS	AD1	MDC0	AE12	VCC1P2
AB21	VSS	AD2	MDIO1	AE13	VCC1P2
AB22	VSS	AD3	RSVDAD3_NC	AE14	VCC1P2
AB23	VSS	AD4	VSS	AE15	VCC1P2
AB24	VSS	AD5	VSS	AE16	VCC1P2
AB25	VCC1P2	AD6	VSS	AE17	VCC1P2
AB26	VSS	AD7	VSS	AE18	VCC1P2
AE19	VCC1P2	AF30	VSS	AH11	VSS
AE20	VCC1P2	AG1	RSVDAG1_NC	AH12	VSS
AE21	VCC1P2	AG2	RBIAS	AH13	VSS
AE22	VCC1P2	AG3	VCC1P2	AH14	VSS
AE23	VSS	AG4	VCC1P2	AH15	VCC1P2
AE24	VSS	AG5	VCC1P2	AH16	VCC1P2
AE25	RSVDAE25_NC	AG6	VCC1P2	AH17	VSS
AE26	VSS	AG7	VCC1P2	AH18	VSS
AE27	VSS	AG8	VCC1P2	AH19	VSS
AE28	VSS	AG9	VCC1P2	AH20	VSS
AE29	PET_1_P	AG10	VCC1P2	AH21	VSS
AE30	PET_1_N	AG11	VCC1P2	AH22	VSS
AF1	RSENSE	AG12	VCC1P2	AH23	VSS
AF2	VSS	AG13	VCC1P2	AH24	VSS
AF3	VSS	AG14	VCC1P2	AH25	VSS
AF4	VSS	AG15	VCC1P2	AH26	VSS
AF5	VSS	AG16	VCC1P2	AH27	VSS
AF6	VCC1P2	AG17	VCC1P2	AH28	SMBD
AF7	VSS	AG18	VCC1P2	AH29	PET_0_P
AF8	VSS	AG19	VCC1P2	AH30	PET_0_N
AF9	VSS	AG20	VCC1P2	AJ1	VSS

Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
AF10	VSS	AG21	VCC1P2	AJ2	VSS
AF11	VSS	AG22	VCC1P2	AJ3	REFCLKIN_N
AF12	VSS	AG23	VCC1P2	AJ4	VSS
AF13	VSS	AG24	VCC1P2	AJ5	TX1_L3_P
AF14	VSS	AG25	VSS	AJ6	TX1_L2_P
AF15	VCC1P2	AG26	VSS	AJ7	TX1_L1_P
AF16	VCC1P2	AG27	VSS	AJ8	TX1_L0_P
AF17	VSS	AG28	PE_WAKE_N	AJ9	VSS
AF18	VSS	AG29	PER_0_P	AJ10	RX1_L3_P
AF19	VSS	AG30	PER_0_N	AJ11	RX1_L2_P
AF20	VSS	AH1	VCC1P2	AJ12	RX1_L1_P
AF21	VSS	AH2	VCC1P2	AJ13	RX1_L0_P
AF22	VSS	AH3	VSS	AJ14	VSS
AF23	VSS	AH4	VSS	AJ15	VCC1P2
AF24	VSS	AH5	VSS	AJ16	VCC1P2
AF25	VSS	AH6	VSS	AJ17	VSS
AF26	VSS	AH7	VSS	AJ18	TX0_L3_P
AF27	VSS	AH8	VSS	AJ19	TX0_L2_P
AF28	VSS	AH9	VSS	AJ20	TX0_L1_P
AF29	VSS	AH10	VSS	AJ21	TX0_L0_P
AJ22	VSS	AK5	TX1_L3_N	AK19	TX0_L2_N
AJ23	RX0_L3_P	AK6	TX1_L2_N	AK20	TX0_L1_N
AJ24	RX0_L2_P	AK7	TX1_L1_N	AK21	TX0_L0_N
AJ25	RX0_L1_P	AK8	TX1_L0_N	AK22	VSS
AJ26	RX0_L0_P	AK9	VSS	AK23	RX0_L3_N
AJ27	SMBCLK	AK10	RX1_L3_N	AK24	RX0_L2_N
AJ28	PE_CLK_P	AK11	RX1_L2_N	AK25	RX0_L1_N
AJ29	VSS	AK12	RX1_L1_N	AK26	RX0_L0_N
AJ30	VSS	AK13	RX1_L0_N	AK27	PE_RST_N
AK1	VSS	AK14	(blank)	AK28	PE_CLK_N

Pin Name	Signal Name	Pin Name	Signal Name	Pin Name	Signal Name
AK2	VSS	AK15	(blank)	AK29	VSS
AK3	REFCLKIN_P	AK17	(blank)	AK30	VSS
AK4	VSS	AK18	TX0_L3_N	AA4	VSS

2.13 Internal/External Pull-Up/Pull-Down Specifications

Table 2-16 and Table 2-17 list internal/external pull-up/pull-down resistor values and whether or not they are activated in the different device states.

For more details about the internal/external pull-up/pull-down requirements, refer to the Intel® 82598 10 GbE Controller board layout/schematic checklists (not included in this datasheet) as well as the reference schematics and design guidelines described later in this datasheet.

Table 2-16. Internal and External Pull-Up and Pull-Down Values

	Min	Nominal	Max	Units
Pull-up (internal)	2.7	5	8.6	KΩ
Pull-up (external, recommended)	3.3		10	KΩ
Pull-down (external, recommended)	100		470	Ω

The 82598 states are defined as follows:

Power-up = while 3.3 V dc is stable, but not 1.2 V dc

Active = normal mode (not power up nor disable)

Table 2-17. Internal/External Pull-Ups/Pull-Downs

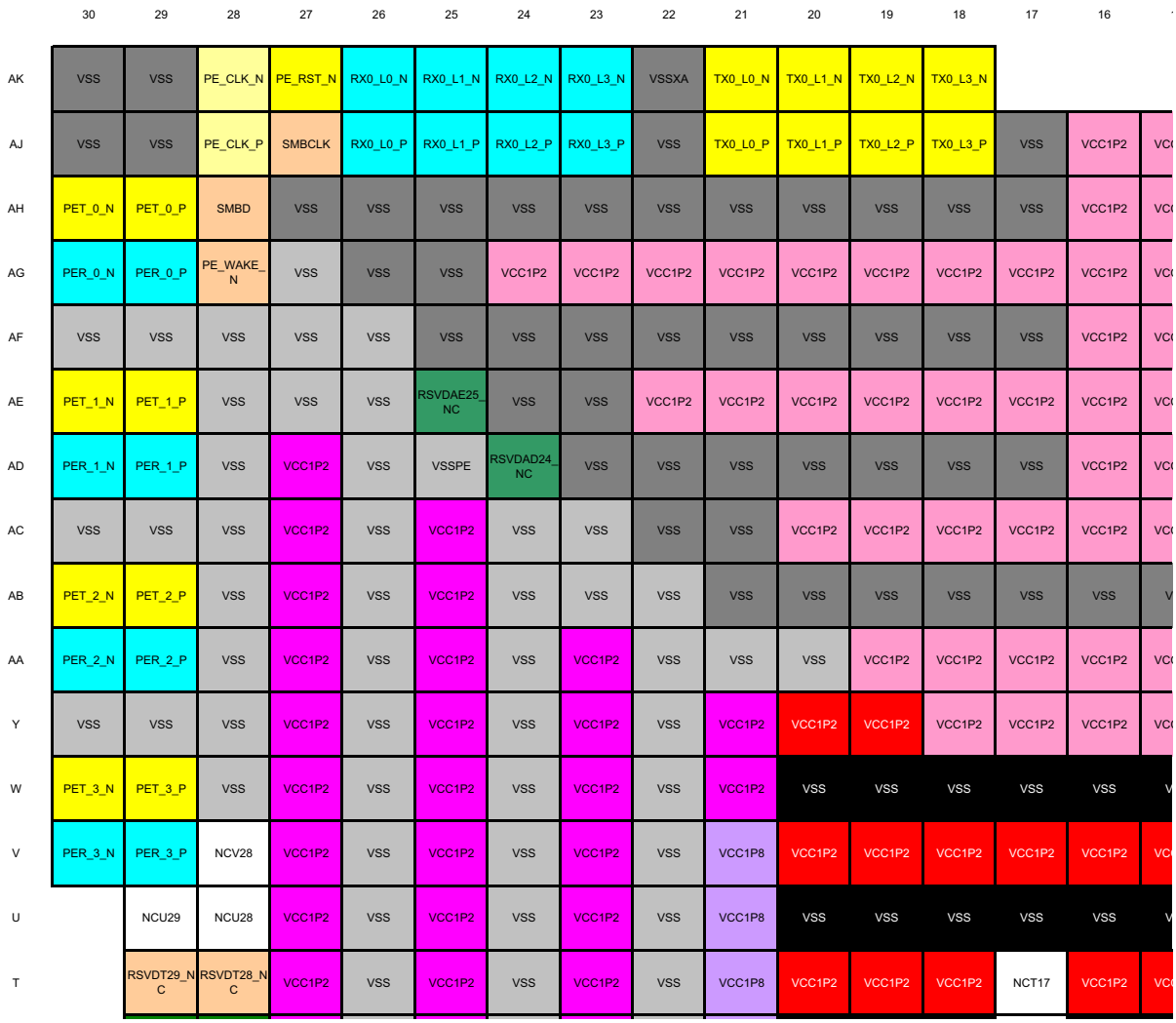
Pin Name	Power Up		Active	
	Internal Pull-Up	Comment	Internal Pull-Up	Comment
EE_DI	Y		N	
EE_DO	Y		Y	
EE_SK	Y		N	
EE_CS_N	Y	External pull-up	N	External pull-up
FLSH_SI	Y		N	
FLSH_SO	Y		Y	
FLSH_SCK	Y		N	

FLSH_CE_N	Y	External pull-up	N	External pull-up
Pin Name	**Power Up**		**Active**	
	Internal Pull-Up	**Comment**	**Internal Pull-Up**	**Comment**
SMBCLK	N	External pull-up	N	External pull-up
SMBD	N	External pull-up	N	External pull-up
SMBALRT_N	N	External pull-up	N	External pull-up
NCSI_CLK_IN	N		N	External pull-down in non NC-SI
NCSI_CRS_DV	Y		N	External pull-down in non NC-SI
NCSI_RXD_0	Y		N	External pull-up in non NC-SI
NCSI_RXD_1	Y		N	External pull-up in non NC-SI
NCSI_TX_EN	N	HiZ	N	External pull-down in non NC-SI
NCSI_TXD_0	N	HiZ	N	External pull-up in non NC-SI
NCSI_TXD_1	N	HiZ	N	External pull-up in non NC-SI
MDIO[0]	Y	External pull-up	N	External pull-up
MDC[0]	Y		N	
MDIO[1]	Y	External pull-up	N	External pull-up
MDC[1]	Y		N	
SDP0[5:0]	Y		Y	
SDP0[7:6]	Y	External pull-up	N	External pull-up
SDP1[5:0]	Y		Y	
SDP1[7:6]	Y	External pull-up	N	External pull-up
LED0[3:0]	Y		N	
LED1[3:0]	Y		N	
LAN_PWR_GOOD	Y		Y	
AUX_PWR	Y		N	External pull-up (if present) External pull-down (not present)
LAN0_DIS_N	Y		Y	
LAN1_DIS_N	Y		Y	
MAIN_PWR_OK	Y		N	
JTCK	Y	External pull-down	N	External pull-down
JTDI	Y		N	External pull-up

JTDO	Y	External pull-up	N	External pull-up
Pin Name	**Power Up**		**Active**	
	Internal Pull-Up	**Comment**	**Internal Pull-Up**	**Comment**
JTMS	Y		N	External pull-up
JRST_N	Y	External pull-down	Y	External pull-down
PE_RST_N	Y		N	
PE_WAKE_N	N	External pull-up	N	External pull-up
POR_BYPASS	Y	External pull-up (if bypassed) External pull-down (if not bypassed)	N	External pull-up (if bypassed) External pull-down (if not bypassed)
PHY0_PWRDN_N	Y		N	
PHY1_PWRDN_N	Y		N	
DEV_PWRDN_N	Y		N	
RSVDE1_VSS	Y		N	External pull-down
RSVDJ1_VSS	Y		N	External pull-down

2.14 Pin Assignments (Ball Out)

This section shows the pin assignments.



	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
AK	VSS	VSS	PE_CLK_N	PE_RST_N	RX0_L0_N	RX0_L1_N	RX0_L2_N	RX0_L3_N	VSSXA	TX0_L0_N	TX0_L1_N	TX0_L2_N	TX0_L3_N			
AJ	VSS	VSS	PE_CLK_P	SMBCLK	RX0_L0_P	RX0_L1_P	RX0_L2_P	RX0_L3_P	VSS	TX0_L0_P	TX0_L1_P	TX0_L2_P	TX0_L3_P	VSS	VCC1P2	VC
AH	PET_0_N	PET_0_P	SMBD	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VCC1P2	VC
AG	PER_0_N	PER_0_P	PE_WAKE_N	VSS	VSS	VSS	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VC
AF	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VCC1P2	VC
AE	PET_1_N	PET_1_P	VSS	VSS	VSS	RSVDAE25_NC	VSS	VSS	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VC
AD	PER_1_N	PER_1_P	VSS	VCC1P2	VSS	VSSPE	RSVDAD24_NC	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VCC1P2	VC
AC	VSS	VSS	VSS	VCC1P2	VSS	VCC1P2	VSS	VSS	VSS	VSS	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VC
AB	PET_2_N	PET_2_P	VSS	VCC1P2	VSS	VCC1P2	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	V
AA	PER_2_N	PER_2_P	VSS	VCC1P2	VSS	VCC1P2	VSS	VCC1P2	VSS	VSS	VSS	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VC
Y	VSS	VSS	VSS	VCC1P2	VSS	VCC1P2	VSS	VCC1P2	VSS	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VC
W	PET_3_N	PET_3_P	VSS	VCC1P2	VSS	VCC1P2	VSS	VCC1P2	VSS	VCC1P2	VSS	VSS	VSS	VSS	VSS	V
V	PER_3_N	PER_3_P	NCV28	VCC1P2	VSS	VCC1P2	VSS	VCC1P2	VSS	VCC1P8	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VCC1P2	VC
U		NCU29	NCU28	VCC1P2	VSS	VCC1P2	VSS	VCC1P2	VSS	VCC1P8	VSS	VSS	VSS	VSS	VSS	V
T		RSVDT29_NC	RSVDT28_NC	VCC1P2	VSS	VCC1P2	VSS	VCC1P2	VSS	VCC1P8	VCC1P2	VCC1P2	VCC1P2	NCT17	VCC1P2	VC

Figure 2-1. Pin Map, Upper Left



Figure 2-2. Pin Assignments, Upper Right



Figure 2-3. Pin Assignments, Lower Left



Figure 2-4. Pin Assignments, Lower Right

§ §



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

3. Functional Description

3.1 Interconnects

3.1.1 PCIe

PCIe defines a set of requirements that address the majority of the targeted application classes. Higher-end application requirements (Enterprise class servers and high-end communication platforms) are addressed by advanced extensions.

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

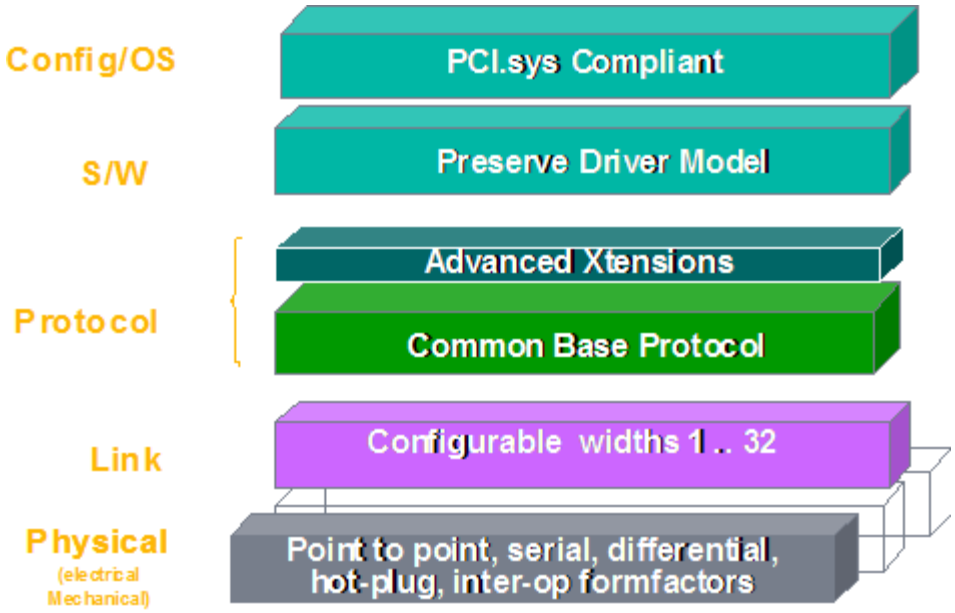


Figure 3-1. PCIe Stack Structure

The PCIe 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-clocked such that no dedicated clock signals are required. The bandwidth increases in direct proportion with frequency.

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

The common base protocol uses split transactions along with several mechanisms to eliminate wait states and to optimize re-ordering transactions to 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 (for example, PCI addressing model):
  - 32-bit memory address space to enable a compact packet header (must be used to access addresses below 4 Gb)
  - 64-bit memory address space using an extended packet header
- Transaction layer mechanisms:
  - PCI-X style relaxed ordering
  - Optimizations for no-snoop transactions
- Credit-based flow control
- Packet sizes/formats:
  - Maximum packet size supports 128-byte and 256-byte data payload
  - Maximum read request size: 256 bytes
- Reset/initialization:
  - Frequency/width/profile negotiation performed by hardware
- Data integrity support:
  - Using CRC-32 for Transaction layer Packets (TLP)
    - Link Layer Retry (LLR) for recovery following error detection:
  - Using CRC-16 for Link Layer (LL) 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 (Interrupts, etc.)
  - System-level power management supported via messages
- Power management:
  - Full support for PCIm
  - Wake capability from D3cold state
  - Compliant with ACPI, PCIm software model
  - Active state power management
- Support for PCIe v2.0 (2.5GT/s):
  - Support for completion time out
  - Support for additional registers in the PCIe capability structure

3.1.1.1.1 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: PCIe v2.0 (2.5GT/s).
- Interface width of x8, x4, x2 or x1.
- DFT and DFM support for high-volume manufacturing

3.1.1.1.2 Advanced Extensions

PCIe defines a set of optional features to enhance platform capabilities for specific modes. The 82598 supports the following optional features:

- Extended Error Reporting – Messaging support to communicate multiple types/severity of errors
- Device Serial Number
- Completion timeout

3.1.1.2 General Functionality

3.1.1.2.1 Native/Legacy

All 82598 PCI functions are native PCIe functions.

3.1.1.2.2 Locked Transactions

The 82598 does not support locked requests as a target or a master.

3.1.1.2.3 End-to-End CRC (ECRC)

This function is not supported by the 82598.

3.1.1.3 Host Interface

PCIe device numbers identify logical devices within the physical device (the 82598 is a physical device). The 82598 implements a single logical device with two separate PCI functions: LAN 0 and LAN 1. The device number is captured from each Type 0 configuration write transaction.

Each 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.

3.1.1.3.1 Tag ID Allocation

Tag IDs are allocated differently for read and write functions.

1. Tag ID allocation for read accesses. The Tag ID is used by hardware in order to be able to forward the read data to the required internal client.

TAG ID	Description
0x0	Data Request 0x0
0x1	Data Request 0x1
0x2	Data Request 0x2
0x3	Data Request 0x3
0x4	Data Request 0x4
0x5	Data Request 0x5
0x6	Data Request 0x6
0x7	Data Request 0x7
0x8	Data Request 0x8
0x9	Data Request 0x9
0xA	Data Request 0xA
0xB	Data Request 0xB
0xC	Data Request 0xC
0xD	Data Request 0xD
0xE	Data Request 0xE
0xF	Data Request 0xF
0x10	Tx Descriptor 0
0x11	Tx Descriptor 1
0x12	Tx Descriptor 2
0x13	Tx Descriptor 3
0x14	Tx Descriptor 4
0x15	Tx Descriptor 5
0x16	Tx Descriptor 6
0x17	Tx Descriptor 7
0x18	Rx Descriptor 0
0x19	Rx Descriptor 1
0x1A	Rx Descriptor 2

TAG ID	Description
0x1B	Rx Descriptor 3
0x1C:0x1F	Reserved

2. TAG ID Allocation for Write Transactions. Request tag allocation depends on these system parameters:
   — DCA supported/not supported in the system
   — DCA enabled/disabled in the command line
   — System type (chipset)
   — CPU ID

The following cases provide usage examples.

Case 1 – DCA Disabled in the System:

The following table lists the write requests tags.

Tag ID	Description
2	Write Back (WB) descriptor Tx /WB head.
4	WB descriptor Rx.
6	Write data.

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

- Fast Side Bus (FSB) platforms – If DCA is disabled for the request, the tags allocation is similar to the case where DCA is disabled in the system.
- CSI platforms – All write requests have the tag of 0x00.

Case 3 – DCA Enabled in the System, DCA Enabled for the Request

- FSB Platforms:
  — Tags are according to the lowest bits of the CPU_ID field.
  — Request tag = {CPU ID [3:0], 1111b}.
- CSI Platforms:
  — Tags are according to the CPU ID.
  — Request tag = CPU ID.

3.1.1.3.2 Completion Timeout Mechanism

In any split transaction protocol, a risk is associated with the failure of a requester to receive an expected completion. To enable requesters to attempt recovery, a completion timeout mechanism is defined. The completion timeout mechanism is activated for each request that requires completions when the request is transmitted. The PCIe v2.0 (2.5 GT/s) specification requires that:

- The completion timeout timer should not expire in less than 10 ms.
- The completion timeout timer must expire if a request is not completed within 50 ms.
- However, some platforms experience completion latencies longer than 50 ms (in some cases up to seconds). The 82598 provides a programmable range for the completion timeout, as well as the ability to disable the completion timeout. PCIe v2.0 (2.5 GT/s) specification defines that completion timeout is programmed through an extension of the PCIe capability structure.

The 82598 controls the following aspects of completion timeout:

- Disabling or enabling completion timeout
- Disabling or enabling resending a request on completion timeout
- A programmable range of timeout values

Programming the behavior of completion timeout is done differently depending on whether capability structure version 0x1 or capability structure version 0x2 (future extension) is enabled. Table 3-1 lists the behavior.

Table 3-1. Completion Timeout Programming

Capability	Capability Structure Version = 0x1	Capability Structure Version = 0x2
Completion Timeout Enabling	Loaded from the EEPROM into a CSR bit.	Controlled through PCI configuration. Visible through a read-only CSR bit.
Resend Request Enable	Loaded from the EEPROM into a CSR bit.	Loaded from the EEPROM into a read-only CSR bit.
Completion Timeout Period	Loaded from the EEPROM into a CSR bit.	Controlled through PCI configuration. Visible through a read-only CSR bit.

3.1.1.3.2.1 Completion Timeout Enable

- Version = 0x1 – Loaded from the *Completion Timeout Disable* bit in the EEPROM into the *Completion_Timeout_Disable* bit in the PCIe Control (GCR) register. The default is *Completion Timeout Enabled*.
- Version = 0x2 – Programmed through the PCI configuration. Visible through the Completion_Timeout_Disable bit in the PCIe Control (GCR) register. The default is: *Completion Timeout Enabled*.

3.1.1.3.2.2 Resend Request Enable

- Version = 0x1 – The *Completion Timeout Resend EEPROM* bit (loaded to the *Completion_Timeout_Resend* bit in the PCIe Control (GCR) register enables resending the request (applies when completion timeout is enabled). The default is to resend a request that timed out.
- Version = 0x2 – same as Rev. 1.1.

3.1.1.3.2.3 Completion Timeout Period

- Version = 0x1 – Loaded from the *Completion Timeout Value* field in the EEPROM to the *Completion_Timeout_Value* bits in the PCIe Control (GCR) register. The following values are supported:
  - 50 µs to 10 ms (default)
  - 10 ms to 250 ms
  - 250 ms to 4 s
  - 4 s to 64 s
- Version = 0x2 – Programmed through the PCI configuration. Visible through the *Completion_Timeout_Value* bits in the PCIe Control (GCR) register. The 82598 supports all four ranges defined by the PCIe ECR:
  - 50 µs to 10 ms
  - 10 ms to 250 ms
  - 250 ms to 4 s
  - 4 s to 64 s

System software programs a range (one of nine possible ranges that sub-divide the previously mentioned four ranges) into the PCI configuration register. The supported sub-ranges are:

- 50 µs to 50 ms (default).
- 50 µs to 100 µs
- 1 ms to 10 ms
- 16 ms to 55 ms
- 65 ms to 210 ms
- 260 ms to 900 ms
- 1 s to 3.5 s
- s to 13 s
- 17 s to 64s

A memory read request for which there are multiple completions is considered complete 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 discard data that was returned prior to expiration.

3.1.1.4 Transaction Layer

The upper layer of the PCIe architecture is the transaction layer. The transaction layer connects to the 82598's core using an implementation-specific protocol. Through this core-to-transaction-layer protocol, application-specific parts of the 82598 interact with the PCIe subsystem and transmits and receives requests to or from a remote PCIe agent, respectively.

3.1.1.4.1 Transaction Types Accepted

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

Transaction Type	FC Type	TX Later Reaction	Hardware Should Keep Data From Original Packet	For Client
Configuration Read Request	NPH	CPLH + CPLD	Requester ID, TAG, Attribute	Configuration Space
Configuration Write Request	NPH + NPD	CPLH	Requester ID, TAG, Attribute	Configuration Space
Memory Read Request	NPH	CPLH + CPLD	Requester ID, TAG, Attribute	CSR
Memory Write Request	PH + PD	–	–	CSR
I/O Read Request	NPH	CPLH + CPLD	Requester ID, TAG, Attribute	CSR
I/O Write Request	NPH + NPD	CPLH	Requester ID, TAG, Attribute	CSR
Read Completions	CPLH + CPLD	–	–	DMA
Message	PH	–	–	Message Unit/INT/ PM/Error Unit

Legend:

- 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

3.1.1.4.1.1 Partial Memory Read and Write Requests

The 82598 has limited support for read and write requests with only part of the byte enable bits set:

- Partial writes with at least one byte enabled are executed as full writes. Any side effect of a full write (such as clear by write) is also applicable to partial writes.
- 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 must be answered 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.

3.1.1.4.2 Transaction Types Initiated

Table 3-3. 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	PH + PD	DMA, MSI/MSI-X
Message	– -	PH	Message Unit/INT/PM/ Error Unit

Note: MAX_PAYLOAD_SIZE is loaded from the EEPROM (either 128 bytes or 256 bytes). Effective MAX_PAYLOAD_SIZE is defined for each PCI function according to the configuration space register for that function.

3.1.1.4.2.1 Data Alignment

Requests must never specify an address/length combination that causes a memory space access to cross a 4 kB boundary. The 82598 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. Consider aligning buffers to a 4 kB boundary in cases where this improves performance.

The general rules for packet alignment are as follows. Note that these apply to all requests (read/write, snoop and no snoop):

1. The length of a single request does not exceed the PCIe limit of MAX_PAYLOAD_SIZE for write and MAX_READ_REQ for read.
2. The length of a single request does not exceed 82598 internal limitations.
3. A single request does not span across different memory pages as noted by the 4 kB boundary alignment above.

If a request can be sent as a single PCIe packet and still meet the general rules for packet alignment, then it is not broken at the cache line boundary but rather sent as a single packet (the chipset might break the request along cache line boundaries, but the 82598 will still benefit from better PCIe use). However, if general rules 1-3 require that the request be broken into two or more packets, then the request will be broken at the cache line boundary.

3.1.1.4.2.2 Multiple Tx Data Read Requests (MULR)

The 82598 supports 16 multiple pipelined requests for transmit data. In general, requests belong to the same packet or to consecutive packets. However, the following restrictions apply:

- 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 above restriction is met. Pipelined requests can belong to the same queue or to separate queues. However, as noted above, all requests for a certain packet are issued (from the same queue) before a request is issued for a different packet (potentially from a different queue).
- The PCIe v2.0 (2.5 GT/s) specification does not insure that completions for separate requests return in-order. Read completions for concurrent requests are not required to return in the order issued. The 82598 handles completions that arrive in any order. Once all completions arrive for a given request, it can issue the next pending read data request.
- The 82598 incorporates a reorder buffer to support re-ordering of completions for all issued requests. Each request/completion can be up to 256 bytes long. The maximum size of a read request is defined as the minimum {256 bytes, Max_Read_Request_Size}.
- In addition to the transmit data requests, the 82598 can issue eight pipelined read requests for Tx descriptors and four pipelined read requests for Rx descriptors. The requests for Tx data, Tx descriptors, and Rx descriptors are independently issued.

3.1.1.5 Messages

3.1.1.5.1 Received Messages

Message packets are special packets that carry a message code. The upstream device transmits special messages to the 82598 by using this mechanism. The transaction layer decodes the message code and responds to the message accordingly.

Table 3-4. Supported Message (as a Receiver)

Message Code [7:0]	Routing r2r1r0	Message	Later Response
0x14	100b	PM_Active_State_NAK	Internal Signal Set
0x19	011b	PME_Turn_Off	Internal Signal Set
0x50	100b	Slot power limit support (has one Dword Data)	Silently Drop
0x7E	010b, 011b,100b	Vendor_defined Type 0 No data	Unsupported Request
0x7E	010b,011b,100b	Vendor_defined Type 0 data	Unsupported Request
0x7F	010b,011b,100b	Vendor_defined Type 1 No data	Silently Drop
0x7F	010b, 011b,100b	Vendor_defined Type 1 data	Silently Drop
0x00	011b	Unlock	Silently Drop

3.1.1.5.2 Transmitted Messages

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

Table 3-5. Initiated Messages

Message code [7:0]	Routing r2r1r0	Message
0x20	100b	Assert INT A
0x21	100b	Assert INT B
0x22	100b	Assert INT C
0x23	100b	Assert INT D
0x24	100b	DE- Assert INT A
0x25	100b	DE- Assert INT B
0x26	100b	DE- Assert INT C
0x27	100b	DE- Assert INT D
0x30	000b	ERR_COR
0x31	000b	ERR_NONFATAL
0x33	000b	ERR_FATAL
0x18	000b	PM_PME
0x1B	101b	PME_TO_Ack

3.1.1.6 Ordering Rules

The 82598 meets the PCIe ordering rules (PCI-X rules) by following the simple device model:

1. 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 (due to no credits), target completions can still proceed (if credits are available).
2. Descriptor/Data Ordering – the device does not proceed with some internal actions until respective data writes have ended on the PCIe link:
3. The 82598 does not update an internal header pointer until the descriptors that the header pointer relates to are written to the PCIe link.
4. The 82598 does not issue a descriptor write until the data that the descriptor relates to is written to the PCIe link.
5. The 82598 might issue the following master read request from each of the following clients:
   a. Rx Descriptor Read (four for each LAN port)
   b. Tx Descriptor Read (eight for each LAN port)
   c. Tx Data Read (up to 16 for each LAN port for manageability)

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

3.1.1.6.1 Out of Order Completion Handling

In a split transaction protocol, using multiple read requests in a multi-processor environment, there is a risk that completions will arrive from the host memory out of order and interleaved. In this case, the 82598 sorts the request completion and transfers them to the Ethernet in the correct order.

3.1.1.7 Transaction Definition and Attributes

3.1.1.7.1 Max Payload Size

The 82598's policy for determining Max Payload Size (MPS) is as follows:

1. Master requests initiated by the 82598 (including completions) limit Max Payload Size to the value defined for the function issuing the request.
2. Target write accesses to the 82598 are accepted only with a size of one Dword or two Dwords. Write accesses in the range of three Dwords (MPS) are flagged as unreliable. Write accesses above MPS are flagged as malformed.

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

The 82598 only supports TC = 0 and VC = 0 (default).

3.1.1.7.3 Relaxed Ordering

The 82598 takes advantage of the relaxed ordering rules in PCIe. By setting the relaxed ordering bit in the packet header, the 82598 enables the system to optimize performance in the following cases:

1. Relaxed ordering for descriptor and data reads – When the 82598 masters 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.
2. Relaxed ordering for receiving data writes – When the 82598 masters receive data writes, it also enables them to bypass each other in the path to system memory because software does not process this data until their associated descriptor writes are done.
3. The 82598 cannot relax ordering for descriptor writes or an MSI write.

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

Relaxed ordering is enabled in the 82598 by clearing the RO_DIS bit in the Extended Device Control (CTRL_EXT) register (0x00018; RW). The actual setting of relaxed ordering is done for LAN traffic by the host through the DCA registers and for headers redirection through an EEPROM setting.

3.1.1.7.4 No Snoop

The 82598 sets the Snoop_Not_Required attribute bit for master data writes. System logic might provide a separate path to system memory for non-coherent traffic. The non-coherent path to system memory provides a higher, more uniform, bandwidth for write requests.

Note: The Snoop Not Required attribute does not alter transaction ordering. Therefore, to achieve the 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.

No snoop is enabled by clearing the NS_DIS bit in the Extended Device Control (CTRL_EXT) register – (0x00018; RW). The actual setting of no snoop is done for LAN traffic by the host through DCA registers and for headers redirection through an EEPROM setting.

3.1.1.7.5 No Snoop and Relaxed Ordering for LAN Traffic

Software might configure no-snoop and relax order attributes for each queue and each type of transaction by setting the respective bits in the DCA_RXCTRL and TCA_TXCTRL registers. Table 3-6 lists the default behavior for the No-Snoop and Relaxed Ordering bits for LAN traffic when I/OAT 2 is enabled.

Table 3-6. LAN Traffic Attributes

Transaction	No Snoop default	Relaxed Ordering default	Comments
Rx Descriptor Read	N	Y	
Rx Descriptor Write-Back	N	N	Read-only. Must never be used for this traffic.
Rx Data Write	Y	Y	See note and the section that follows.
Rx Replicated Header	N	Y	
Tx Descriptor Read	N	Y	
Tx Descriptor Write-Back	N	Y	
Tx Data Write	N	Y	

Note: RX payload no-snoop is also conditioned by the *NSE* bit in the receive descriptor.

3.1.1.7.5.1 No Snoop Option for Payload

Under certain conditions, which occur when I/OAT 2 is enabled, software knows that it is safe to transfer a new packet into a certain buffer without snooping on the FSB. This scenario 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 data movement engine. In this case, software should be able to set a bit in the receive descriptor indicating that the 82598 should perform a no-snoop transfer when it eventually writes a packet to this buffer. When a no-snoop transaction is activated, the TLP header has a no-snoop attribute in the Transaction Descriptor field. This is triggered by the *NSE* bit in the receive descriptor.

3.1.1.8 Flow Control

3.1.1.8.1 82598 Flow Control Rules

The 82598 implements only the default Virtual Channel (VC0). A single set of credits is maintained for VC0.

Table 3-7. Flow Control Credits Allocation

Credit Type	Operations	Number of Credits
Posted Request Header (PH)	Target Write (1 unit) Message (1 unit)	8 units (to enable concurrent accesses to both LAN ports).
Posted Request Data (PD)	Target Write (Length/16 bytes = 1) Message (1 unit)	MAX_PAYLOAD_SIZE/16.
Non-Posted Request Header (NPH)	Target Read (1 unit) Configuration Read (1 unit) Configuration Write (1 unit)	4 units (to enable concurrent target accesses to both LAN ports).
Non-Posted Request Data (NPD)	Configuration Write (1 unit)	4 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 82598 maintains two credits for NPD at any given time. It increments the credit by one after a credit is consumed and sends an UpdateFC packet as soon as possible. UpdateFC packets are scheduled immediately after a resource is available.
- The 82598 provides two credits for PH (such as, for two concurrent target writes) and two credits for NPH (such as, for two concurrent target reads). UpdateFC packets are scheduled immediately after a resource is available.
- The 82598 follows the PCIe recommendations for frequency of UpdateFC FCPs.

3.1.1.8.2 Upstream Flow Control Tracking

The 82598 issues a master transaction only when the required flow control credits are available. Credits are tracked for posted, non-posted, and completions (the later to operate against a switch).

3.1.1.8.3 Flow Control Update Frequency

In all cases UpdateFC packets are scheduled immediately after a resource is available.

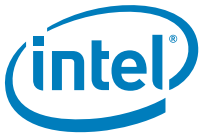
When the Link is in the L0 or L0s link state, Update FCPs for each enabled type of non-infinite flow control credit must be scheduled for transmission at least once every 30 µs (-0% /+50%), except when the *Extended Sync* bit of the Control Link register is set, in which case the limit is 120 µs (-0%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

3.1.1.8.4 Flow Control Timeout Mechanism

The 82598 implements the optional flow control update timeout mechanism.

The mechanism is active 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.

Upon timer expiration, the mechanism instructs the PHY to retrain the link (using the LTSSM Recovery state).



3.1.1.9 Error Forwarding

If a TLP is received with an error-forwarding trailer, the packet is dropped and is not delivered to its destination. The 82598 does not initiate additional master requests for that PCI function until it detects an internal software reset for associated 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.

3.1.1.10 Link Layer

3.1.1.10.1 ACK/NAK Scheme

The 82598 supports two alternative schemes for ACK/NAK rate:

1. ACK/NAK is scheduled for transmission following any TLP.
2. ACK/NAK is scheduled for transmission according to timeouts specified in the PCIe v2.0 (2.5 GT/s) specification.

The PCIe Error Recovery bit (loaded from the EEPROM) determines which of the two schemes is used.

3.1.1.10.2 Supported DLLPs

The following DLLPs are supported by the 82598 as a receiver:

1. ACK
2. NAK
3. PM_Request_Ack
4. InitFC1-P
5. InitFC1-NP
6. InitFC1-Cpl
7. InitFC2-P
8. InitFC2-NP
9. InitFC2-Cpl
10. UpdateFC-P
11. UpdateFC-NP
12. UpdateFC-Cpl

The following DLLPs are supported by the 82598 as a transmitter:

1. ACK
2. NAK
3. PM_Enter_L1
4. PM_Enter_L23
5. InitFC1-P
6. InitFC1-NP
7. InitFC1-Cpl
8. InitFC2-P
9. InitFC2-NP

10. InitFC2-Cpl
11. UpdateFC-P
12. UpdateFC-NP

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

3.1.1.10.3 Transmit EDB Nullifying

In the event of a retrain necessity, 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 the EDB to the packet.

3.1.1.11 PHY

3.1.1.11.1 Link Speed

The 82598 supports PCIe v2.0 (2.5 GT/s).

The 82598 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 82598 supports entering compliance mode at the speed indicated in the Target Link Speed field in the PCIe Link Control 2 register.

3.1.1.11.2 Link Width

- The 82598 supports a maximum link width of x8, x4, x2, or x1 as determined by the EEPROM *Lane_Width* field.

The maximum link width is loaded into the *Maximum Link Width* field of the PCIe Capability register (LCAP[11:6]). The hardware default is the x8 link.

During link configuration, the platform and the 82598 negotiate on a common link width. The link width must be one of the supported PCIe link widths (x1, 2x, x4, x8), such that:

- If Maximum Link Width = x8, then the 82598 negotiates to either x8, x4, x2 or x1[1]
- If Maximum Link Width = x4, then the 82598 negotiates to either x4 or x1
- If Maximum Link Width = x1, then the 82598 only negotiates to x1

The 82598 does not initiate a hardware autonomous link width change and the *Hardware Autonomous Width Disable* bit in the PCIe Link Control register is hardwired to 0b.

3.1.1.11.3 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 5 D5.2. This provides the clear indication of lane polarity inversion.

1. See restriction in Section 3.1.1.11.6.

3.1.1.11.4 L0s Exit Latency

The number of FTS sequences (N_FTS) sent during L0s exit is loaded from the EEPROM into an 8-bit read-only register.

3.1.1.11.5 Lane-to-Lane De-Skew

A multi-lane link can have many sources of lane-to-lane skew. Although symbols are transmitted simultaneously on all lanes, they cannot be expected to arrive at the receiver without lane-to-lane skew. The lane-to-lane skew can include components, which are less than a bit time, bit time units (400 ps for 2.5 Gb), or full symbol time units (4 ns) of skew caused by the retiming repeaters' insert/delete operations. Receivers use TS1 or TS2 or Skip Ordered Sets (SOS) to perform link de-skew functions.

The 82598 supports de-skew of up to five symbols time (20 ns).

3.1.1.11.6 Lane Reversal

The following lane reversal modes are supported:

- Lane configurations x8, x4, x2, and x1
- Lane reversal in x8 and in x4
- Degraded mode (downshift) from x8 to x4 to x2 to x1 and from x4 to x1, with one restriction – if lane reversal is executed in x8, then downshift is only to x1 and not to x4

Figure 3-2 through Figure 3-5 shows the lane downshift in both regular and reversal connections as well as lane connectivity from a system level perspective.

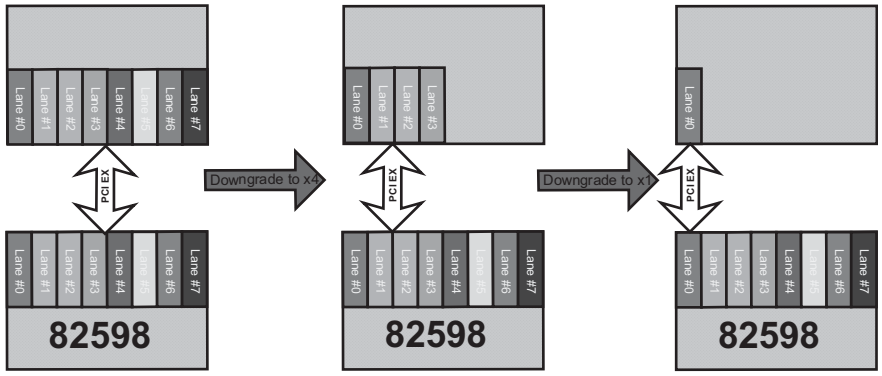


Figure 3-2. Lane Downshift in an x8 Configuration

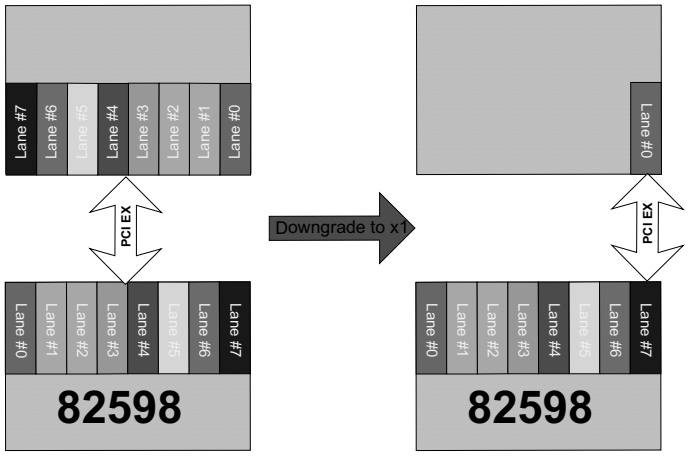


Figure 3-3. Lane Downshift in a Reversal x8 Configuration

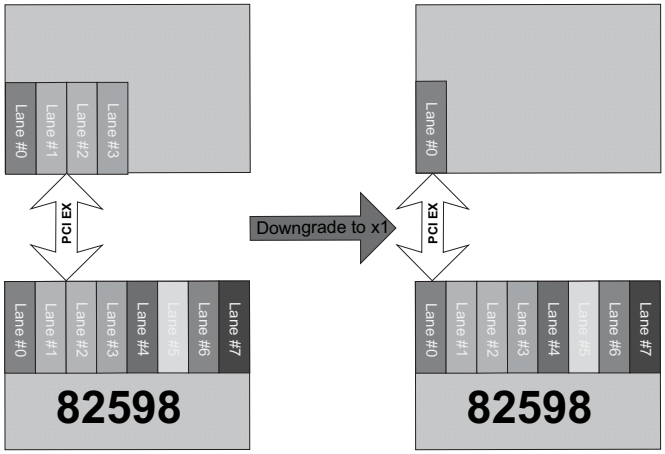


Figure 3-4. Lane Downshift in a x4 Configuration

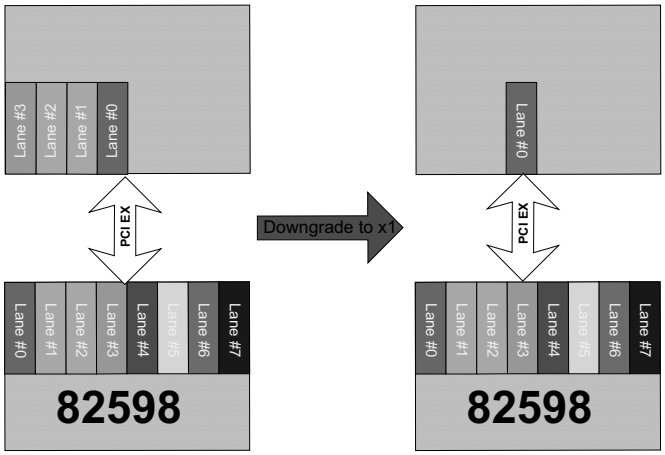


Figure 3-5. Lane Downshift in an x4 Reversal Configuration

The lane reversal feature can be controlled by the EEPROM *Lane Reversal Disable* bit.

3.1.1.11.7 Reset

The PCIe PHY supplies core reset to the 82598. Reset can be caused by the following:

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

3.1.1.11.8 Scrambler Disable

The scrambler/de-scrambler functionality in the 82598 can be eliminated by three mechanisms:

1. Upstream, according to the PCIe v2.0 (2.5 GT/s) specification.
2. EPROM bit – Scram_dis.
3. IBIST JTAG CSR.

3.1.1.12 Error Events and Error Reporting

3.1.1.12.1 General Description

PCIe defines two error reporting paradigms: the baseline capability and the Advanced Error Reporting (AER) capability. Baseline error report 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 82598.

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

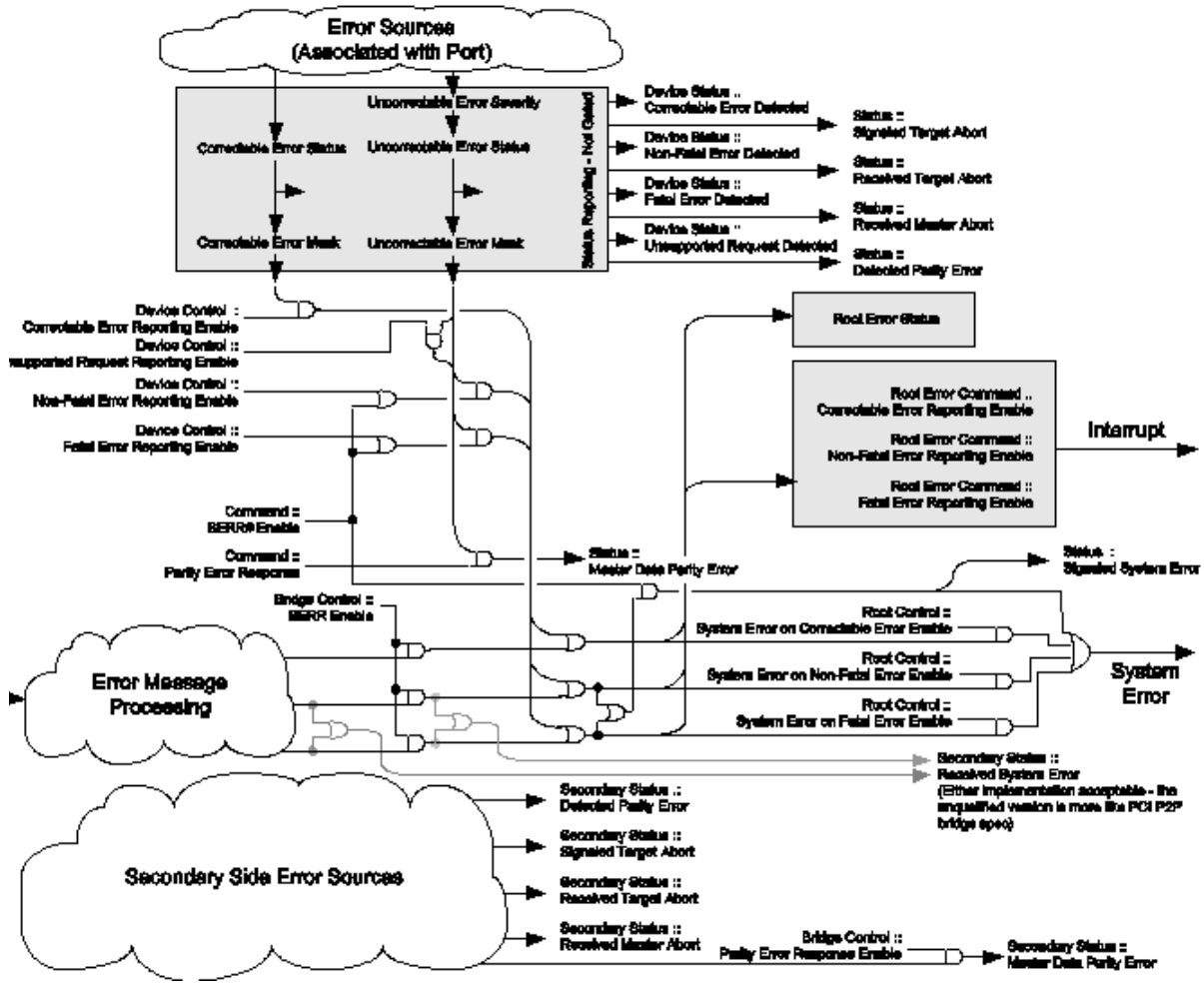


Figure 3-6. Error Reporting Mechanism

3.1.1.12.2 Error Events

Table 3-8 lists the error events identified by the 82598 and the response in terms of logging, reporting, and actions taken. Refer to the PCIe v2.0 (2.5 GT/s) specification for the effect on the PCI Status register.

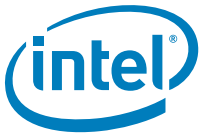


Table 3-8. Response and Reporting of PCIe Error Events

Error Name	Error Events	Default Severity	Action
Physical Layer Errors			
Receiver Error	• 8b/10b Decode Errors • Packet Framing Error	Correctable Send ERR_CORR	TLP to Initiate NAK, Drop Data DLLP to Drop
Data Link Errors			
Bad TLP	• Bad CRC • Not Legal EDB • Wrong Sequence Number	Correctable Send ERR_CORR	TLP to Initiate NAK, Drop Data
Bad DLLP	• Bad CRC	Correctable Send ERR_CORR	DLLP to Drop
Replay timer timeout	• REPLAY_TIMER expiration	Correctable Send ERR_CORR	Follow LL Rules
REPLAY NUM Rollover	• REPLAY NUM Rollover	Correctable Send ERR_CORR	Follow LL Rules
Data Link Layer Protocol Error	• Violations of Flow Control Initialization Protocol	Uncorrectable Send ERR_FATAL	
TLP Errors			
Poisoned TLP Received	• TLP With Error Forwarding	Uncorrectable ERR_NONFATAL Log Header	If Poisoned Completion, No More Requests From This Client
Unsupported Request (UR)	• Wrong Config Access • MRdLk • Config Request Type1 • Unsupported Vendor Defined Type 0 Message • Not Valid MSG Code • Not Supported TLP Type • Wrong Function Number • Wrong TC • Received Target Access With Data Size >64 bits • Received TLP Outside Address Range	Uncorrectable ERR_NONFATAL Log header	Send Completion With UR
Completion Timeout	• Completion Timeout Timer Expired	Uncorrectable ERR_NONFATAL	Send the Read Request Again
Completer Abort	• Attempts to write to the Flash device when writes are disabled (FWE=10b)	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

Error Name	Error Events	Default Severity	Action
Flow Control Protocol Error	• Minimum Initial Flow Control Advertisements • Flow Control Update for Infinite Credit Advertisement	Uncorrectable. ERR_FATAL	Receiver Behavior is Undefined
Malformed TLP (MP)	• Data Payload Exceed Max_Payload_Size • Received TLP Data Size Does Not Match Length Field • TD field value does not correspond with the observed size • Byte Enables Violations • PM Messages That Don't Use TC0 • Usage of Unsupported VC	Uncorrectable ERR_FATAL Log Header	Drop the Packet, Free FC Credits
Completion with Unsuccessful Completion Status		No Action (already done by originator of completion)	Free FC Credits

3.1.1.12.3 Error Pollution

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

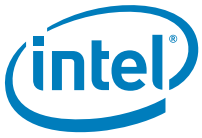
3.1.1.12.4 Completion With Unsuccessful Completion Status

A completion with unsuccessful completion status is dropped and not delivered to its destination. The request that corresponds to the unsuccessful completion is retried by sending a new request for the data.

3.1.1.12.5 Error Reporting Changes

The PCIe v2.0 (2.5 GT/s) specification defines two changes to advanced error reporting. The *Role-Based Error Reporting* bit in the Device Capabilities register is set to 1b to indicate that these changes are supported:

- Setting the *SERR# Enable* bit in the PCI Command register 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.
- Changes in the response to some uncorrectable non-fatal errors detected in non-posted requests to the 82598. These are called Advisory Non-Fatal Error cases. For the errors listed, the following is defined:
  - 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.



— 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 previous unserviced error.

— 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 requestor of a non-posted request times out while waiting for the associated completion, the requestor 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 it must signal the error (if enabled) with an uncorrectable error message if no further recovery attempt is 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.
- 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.

3.1.1.13 Performance Monitoring

The 82598 incorporates PCIe performance monitoring counters to provide common capabilities to evaluate performance. The device implements four 32-bit counters to correlate between concurrent measurements of events as well as the sample delay and interval timers. The four 32-bit counters can also operate in 64-bit mode to count long intervals or payloads.

The list of events supported by the 82598 and the counters *Control* bits are described in the PCIe Register section (see Section 4.).

3.1.1.14 Configuration Registers

3.1.1.14.1 PCI Compatibility

PCIe is compatible with existing deployed PCI software. PCIe hardware implementations conform to the following requirements:

1. All devices are required to support deployed PCI software and must be enumerable as part of a tree through PCI device enumeration mechanisms.
2. Devices must not require resources (such as address decode ranges and interrupts) beyond those claimed by PCI resources for operation of existing deployed PCI software.
3. Devices in their default operating state must confirm to PCI ordering and cache coherency rules from a software viewpoint.
4. PCIe devices must conform to the PCI power management specification and must not require any register programming for PCI-compatible power management beyond those available through PCI power management capability 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 PCIe v2.0 (2.5 GT/s) 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 82598 is a multi-function device with the following functions:
- LAN 0
- LAN 1

Different parameters affect how LAN functions are exposed on PCIe.

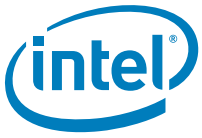
All functions contain the following regions of the PCI configuration space:
- Mandatory PCI configuration registers
- Power management capabilities
- MSI capabilities
- PCIe extended capabilities

3.1.1.14.2 Configuration Sharing Among PCI Functions

The 82598 contains a single physical PCIe core interface. It is designed so that each logical LAN device (LAN 0, LAN 1) appears as a distinct function implementing, amongst other registers, PCIe device header space as listed in Table 3-9.

Table 3-9. PCIe Device Header Space Map

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x0	Device ID		Vendor ID	
0x4	Status Register		Command Register	
0x8	Class Code (0x020000)			Revision ID (0x03)
0xC	Reserved (0x00)	Header Type (0x00)	Latency Timer	Cache Line Size
0x10	Base Address 0			
0x14	Base Address 1			
0x18	Base Address 2			
0x1C	Base Address 3			
0x20	Base Address 4			
0x24	Base Address 5			
0x28	CardBus CIS Pointer (not used)			
0x2C	Subsystem ID		Subsystem Vendor ID	
0x30	Expansion ROM Base Address			
0x34	Reserved			Cap_Ptr
0x38	Reserved			
0x3C	Max_Latency (0x00)	Min_Grant (0xff)	Interrupt Pin (0x01 or 0x02)	Interrupt Line (0x00)



Many of the fields of the PCIe header space contain hardware default values that are either fixed or can be overridden using an EEPROM, but might not be independently specified for each logical LAN device. The fields listed in the table below are common to both LAN devices.

Table 3-10. Fields Common to LAN 0, LAN 1

Vendor ID	The Vendor ID of the 82598 is specified to a single value of 0x8086. The value is reflected identically for both LAN devices.
Revision	The revision number of the 82598 is reflected identically for both LAN devices.
Header Type	This field indicates if a device is single function or multifunction. The value reflected in this field is reflected identically for both LAN devices, but the actual value reflected depends on LAN disable configuration. When both the 82598 LAN ports are enabled, both PCIe headers return 0x80 in this field, acknowledging being part of a multi-function device. LAN 0 exists as device function 0, while LAN 1 exists as device function 1. If function 1 is disabled, then only a single-function device is indicated (this field returns a value of 0x00) and the LAN exists as device function 0.
Subsystem ID	The subsystem ID of the 82598 can be specified via an EEPROM, but only a single value can be specified. The value is reflected identically for both LAN devices.
Subsystem Vendor ID	The subsystem Vendor ID of the 82598 can be specified via an EEPROM, but only a single value can be specified. The value is reflected identically for both LAN devices.
Class Code, Cap_Ptr, Max Latency, Min Grant	These fields reflect fixed values that are constant values reflected for both LAN devices.

The following fields are implemented uniquely for each LAN device.

Table 3-11. Unique Fields

Device ID	The device ID reflected for each LAN device can be independently specified via an EEPROM.
Command, Status	Each LAN device implements its own Command/Status registers.
Latency Timer, Cache Line Size	Each LAN device implements these registers uniquely. The system should program these fields identically for each LAN to ensure consistent behavior and performance of each device.
Memory BAR, Flash BAR, IO BAR, Expansion ROM BAR	Each LAN device implements its own Base Address registers, enabling each device to claim its own address region(s).
Interrupt Pin	Each LAN device independently indicates which interrupt pin (INTA# or INTB#) is used by that device's MAC to signal system interrupts. The value for each LAN device can be independently specified via an EEPROM, but only if both LAN devices are enabled.

3.1.1.14.3 Mandatory PCI Configuration Registers

The PCI configuration registers map is depicted below. Refer to the detailed descriptions for registers loaded from the EEPROM at initialization. Initialization values of the configuration registers are marked in parenthesis. Notation:

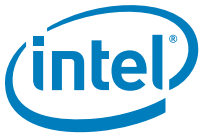
- Dotted – Fields are identical to all functions
- Light-blue – Read-only fields
- Magenta – Hardcoded
- Configuration registers are assigned one of the attributes listed in Table 3-12.

Table 3-12. Attributes of Configuration Registers

R/W	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-1b-to-clear status register; writing a 0b to R/W1C bits has no effect.
ROS	Read-only register with sticky bits. Register bits are read-only and cannot be altered by software. Bits are not cleared by reset and can only be reset with the PWRGOOD signal. Devices that consume AUX power are not allowed to reset sticky bits when AUX power consumption (either via AUX power or PME Enable) is enabled.
RWS	Read-write register 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-1b-to-clear status register. Register bits indicate status when read, a set bit, indicating a status event, can be cleared by writing a 1b to it. Writing a 0b to 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 EEPROM. Bits are read-only after initialization and can only be reset (for write-once by firmware) with the PWRGOOD signal.
RsvdP	Reserved and preserved. Reserved for future read-write implementations; software must preserve value read for writes to these bits.
RsvdZ	Reserved and zero. Reserved for future R/W1C implementations; software must use 0b for writes to these bits.

Table 3-13. PCI-Compatible Configuration Registers

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x0	Device ID		Vendor ID (0x8086)	
0x4	Status Register (0x0010)		Command Register (0x0000)	
0x8	Class Code (0x020000, 0x010185, 0x070002, 0x0C0701)			Revision ID (0x03)
0xC	Reserved (0x00)	Header Type (0x00 | 0x80)	Latency Timer (0x00)	Cache Line Size (0x10)
0x10	Base Address 0			
0x14	Base Address 1			
0x18	Base Address 2			
0x1C	Base Address 3			



0x20	Base Address 4			
0x24	Base Address 5			
0x28	Cardbus CIS Pointer (0x00000000)			
0x2C	Subsystem ID (0x0000)		Subsystem Vendor ID (0x8086)	
0x30	Expansion ROM Base Address			
0x34	Reserved (0x000000)			Cap_Ptr (0x40)
0x38	Reserved (0x00000000)			
0x3C	Max_Latency (0x00)	Min_Grant (0x00)	Interrupt Pin (0x01)	Interrupt Line (0x00)

Interpretation of the various registers in the 82598 are described in the sections that follow.

Vendor ID – This is a read-only register that has the same value for all PCI functions. It identifies unique Intel products.

Device ID – This is a read-only register. It has the same value for the two LAN functions. This field identifies unique 82598 functions. The field can be auto-loaded from the EEPROM during initialization with the following default values:

PCI Function	Default Value	Meaning
LAN 0	10B6	Dual Port 10G/1G Ethernet controller x8 PCIe.
LAN 1	10B6	Dual Port 10G/1G Ethernet controller x8 PCIe.

Command Reg. These are read-write registers. Shaded bits are not used by this implementation and are set to 0b. Each function has its own Command register.

Bit(s)	Initial Value	Description
0	0b	I/O Access Enable.
1	0b	Memory Access Enable.
2	0b	Enable Mastering. LAN 0 read-write field. LAN 1 read-write field.
3	0b	Special Cycle Monitoring – Hardwired to 0b.
4	0b	MWI Enable – Hardwired to 0b.
5	0b	Palette Snoop Enable – Hardwired to 0b.
6	0b	Parity Error Response.
7	0b	Wait Cycle Enable – Hardwired to 0b.

8	0b	SERR# Enable.
9	0b	Fast Back-to-Back Enable – Hardwired to 0b.
10	0b	Interrupt Disable.[1]
15:11	0b	Reserved.

1. 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.

Status Register – Shaded bits are not used by this implementation and are set to 0b. Each function has its own Status register. Unless explicitly specified, entries are the same for all functions.

Bits	Initial Value	R/W	Description
2:0	0b		Reserved.
3	0b	RO	Interrupt Status.[1]
4	1b	RO	New Capabilities. Indicates that a device implements extended capabilities. The 82598 sets this bit and implements a capabilities list to indicate that it supports PCI power management MSIs and PCIe extensions.
5	0b		66 MHz Capable. Hardwired to 0b.
6	0b		Reserved.
7	0b		Fast Back-to-Back Capable. Hardwired to 0b.
8	0b	R/W1C	Data Parity Reported.
10:9	00b		DEVSEL Timing. Hardwired to 0b.
11	0b	R/W1C	Signaled Target Abort.
12	0b	R/W1C	Received Target Abort
13	0b	R/W1C	Received Master Abort.
14	0b	R/W1C	Signaled System Error.
15	0b	R/W1C	Detected Parity Error.

1. The *Interrupt Status* field is a RO field that indicates that an interrupt message is pending internally to the device.

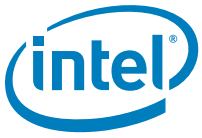
Revision – The default revision ID of this device is 0x00. The value of the rev ID is a logic XOR between the default value and the value in EEPROM word 0x1D. Note that LAN 0 and LAN 1 functions have the same revision ID.

Class Code – The class code is a read-only hard-coded value that identifies the device functionality.

- **LAN 0, LAN 1** – 0x020000 (Ethernet Adapter)

Cache Line Size – This field is implemented by PCIe devices as a read-write field for legacy purposes; it has no PCIe device functionality. Loaded from EEPROM. All functions are initialized to the same value.

Latency Timer – Not used. Hardwired to 0b.



Header Type – This indicates if a device is single- or multi-function. If a single LAN function is the only active one, this field has a value of 0x00 to indicate a single-function device. If other functions are enabled, this field has a value of 0x80 to indicate a multi-function device.

Base Address Registers – The Base Address Registers (or BARs) are used to map register space of various functions. 32-bit addresses are used in one register for each memory mapping window.

Table 3-14. LAN 0 & LAN 1 Functions

BAR	Addr	31:4	3	2:1	0
0	0x10	Memory BAR (R/W – 31:17; 0b – 16:4)	0b	00b	0b
1	0x14	Flash BAR (R/W – 31:23/16; 0b – 22/15:4) Refer to previously mentioned text regarding Flash size.	0b	00b	0b
2	0x18	I/O BAR (R/W – 31:5; 0b – 4:1)		0b	1b
3	0x1C	MSI-X BAR (R/W – 31:14; 0b – 13:4)	0b	00b	0b
4	0x20	Reserved (read as all 0b's)			
5	0x24	Reserved (read as all 0b's)			

All base address registers have the following fields:

Field	Bit(s)	R/W	Initial Value	Description
Mem	0	R	0b for Memory 1b for I/O	0b = Indicates memory space. 1b = Indicates I/O.
Mem Type	2:1	R	00b	Indicates the address space size. 00b = 32-bit
Prefetch Mem	3	R	0b	0b = Non-prefetchable space. 1b = Prefetchable space. The 82598 implements non-prefetchable space since it has read side effects.
Memory Address Space	31:4	R/W	0b	Read-write bits are hardwired to 0b and dependent on memory mapping window sizes. • LAN memory spaces are 128 kB. • LAN Flash spaces can be either 64 kB or up to 8 MB in the power of 2. Mapping window size is set by EEPROM word 0x0F.
IO Address Space	31:2	R/W	0b	Read-write bits are hardwired to 0b and dependent on I/O mapping window sizes. • LAN I/O spaces are 32 bytes

Table 3-15. Memory and IO Mapping

Function	Mapping Window	Mapping Description
LAN 0 LAN 1	Memory BAR 0	The internal registers and memories are accessed as direct memory mapped offsets from the Base Address register. Software can access a Dword or 64 bits.
	Flash BAR 1	The external Flash can be accessed using direct memory mapped offsets from the Flash Base Address register. Software can access byte, word, Dword or 64 bits.
	I/O BAR 2	All internal registers, memories, and Flash can be accessed using I/O operations. There are two 4-byte registers in the IO mapping window: Addr Reg and Data Reg. Software can access byte, word or Dword.
	MSI-X BAR 3	The internal registers and memories are accessed as direct memory mapped offsets from the Base Address register. Software can access a Dword or 64 bits.

3.1.1.14.3.1 Expansion ROM Base Address

This register is used to define the address and size information for boot-time access to optional Flash memory. It is enabled by EEPROM words 0x24 and 0x14 for LAN 0 and LAN 1, respectively. This register returns a zero value for functions without an expansion ROM window.

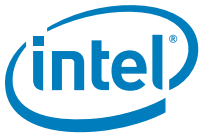
31:11	10:1	0
Expansion Rom BAR (R/W – 31:12316; 0b – 22/15:1) Refer to the previously mentioned text regarding Flash BAR.		En

Field	Bit(s)	R/W	Initial Value	Description
En	0	R/W	0b	1b = Enables expansion ROM access. 0b = Disables expansion ROM access.
Reserved	10:1	R	0b	Always read as 0b. Writes are ignored.
Address	31:11	R/W	0b	Read-write bits are hardwired to 0b and dependent on the memory mapping window size. LAN Expansion ROM spaces can be either 64 kB or up to 8 MB in the power of 2. Mapping window size is set by EEPROM word 0x0F.

Subsystem ID – This value can be loaded automatically from the EEPROM at power up with a default value of 0x0000.

PCI Function	Default Value	EEPROM Address
LAN Functions	0x0000	0x0B

Subsystem Vendor ID – This value can be loaded automatically from the EEPROM at power up or reset. A value of 0x8086 is the default for this field at power up if the EEPROM does not respond or is not programmed. All functions are initialized to the same value.



Cap_Ptr – The Capabilities Pointer field (Cap_Ptr) is an 8-bit field that provides an offset in the 82598's PCI configuration space for the location of the first item in the capabilities linked list. The 82598 sets this bit, and implements a capabilities list to indicate that it supports PCI power management, MSIs, and PCIe extended capabilities. Its value is 0x40, which is the address of the first entry: PCI power management.

Address	Item	Next Pointer
0x40-47	PCI Power Management.	0x50
0x50-5F	Message Signaled Interrupt.	0x60
0x60-6F	Extended Message Signaled Interrupt.	0xA0
0xA0-DB	PCIe Capabilities.	0x00

Interrupt Pin – Read-only register.

- LAN 0 / LAN 1[1]- A value of 0x1/0x2 indicates that this function implements a legacy interrupt on INTA/INTB respectively. Loaded from EEPROM word 0x24/0x14 for LAN 0 and LAN 1, respectively. Refer to the following detail for cases in which LAN port(s) are disabled.

Interrupt Line – Read/write register programmed by software to indicate which of the system interrupt request lines the 82598's interrupt pin is bound to. Refer to the PCI definition for more details.

Max_Lat/Min_Gnt – Not used. Hardwired to 0b.

3.1.1.14.4 PCI Power Management Registers

All fields are reset at full power-up. All fields except PME_En and PME_Status are reset after exiting from the D3cold state. If AUX power is not supplied, the PME_En and PME_Status fields reset after exiting from the D3cold state.

Refer to the detailed description below for registers loaded from the EEPROM at initialization. Initialization values of the Configuration registers are marked in parenthesis.

Notation:

- Dotted – Fields that are identical to all functions
- Light-blue – Read-only fields
- Magenta – Hardcoded and strapping option

Table 3-16. Power Management Register Block

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x40	Power Management Capabilities (PMC)		Next Pointer	Capability ID
0x44	Data	PMCSR_BSE Bridge Support Extensions	Power Management Control/Status Register (PMCSR)	

1. If only a single device/function of the 82598 component is enabled, this value is ignored, and the Interrupt Pin field of the enabled device reports INTA# usage.

The following section describes the register definitions, whether they are required or optional for compliance, and how they are implemented in the 82598.

Capability ID – 1 Byte, Offset 0x40, (RO) – This field equals 0x01 indicating the linked list item as being the PCI Power Management register.

Next Pointer – 1 Byte, Offset 0x41, (RO) – This field provides an offset to the next capability item in the capability list. Its value of0x50 points to MSI capability.

Power Management Capabilities (PMC) – 2 Byte, Offset 0x42, (RO) – This field describes the device functionality during the power management states as listed in Table 3-17. Note that each device function has its own register.

Table 3-17. Power Management Capabilities (PMC)

Bits	Default	R/W	Description
15:11	01001b	RO	PME_Support. This 5-bit field indicates the power states in which the function can assert PME#. Its initial value is loaded from EEPROM word 0x0A. Condition Functionality Values: • No AUX Pwr PME at D0 and $D3_{hot}$ = 01001b • AUX Pwr PME at D0, $D3_{hot}$, and $D3_{cold}$ = 11001b
10	0b	RO	D2_Support – 82598 does not support the D2 state.
9	0b	RO	D1_Support – 82598 does not support the D2 state.
8:6	000b	RO	AUX Current – Required current defined in the Data register.
5	1b	RO	DSI – 82598 requires its device driver to be executed following a transition to the D0 uninitialized state.
4	0b	RO	Reserved.
3	0b	RO	PME_Clock – Disabled. Hardwired to 0b.
2:0	011b	RO	Version – 82598 complies with the PCI PM specification revision 1.2.

Power Management Control/Status Register (PMCSR) – 2 Byte, Offset 0x44, (R/W) – This register is used to control and monitor power management events in the device. Note that each device function has its own PMCSR.

Table 3-18. Power Management Control/Status (PMCSR)

Bits	Default	R/W	Description
15	0b at power up	R/W1C	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	Refer to the value in the Data register description that follows	RO	Data_Scale. This field indicates the scaling factor that's used when interpreting the value of the Data register. For the LAN function, this field equals 01b (indicating 0.1 watt/units) and the Data_Select field is set to 0, 3, 4, 7, (or 8 for function 0). Otherwise, it equals 00b.
12:9	0000b	R/W	Data_Select. This 4-bit field is used to select which data is to be reported through the Data register and Data_Scale field. These bits are writeable only when power management is enabled via the EEPROM.
8	0b at power up	R/W	PME_En. If power management is enabled in the EEPROM, writing a 1b to this register enables wake up. If power management is disabled in the EEPROM, writing a 1b to this bit has no effect and does not set the bit to 1b.
7:4	0000b	RO	Reserved. 82598 returns a value of 000000b for this field.
3	0b	RO	No_Soft_Reset. This bit is always set to 0b to indicate that 82598 performs an internal reset upon transitioning from D3hot to D0 via software control of the PowerState bits. Configuration context is lost when performing the soft reset. Upon transition from the D3hot to the D0 state, a full re-initialization sequence is needed to return the 82598 to the D0 Initialized state.
2	0b	RO	Reserved for PCIe.
1:0	00b	R/W	PowerState. This field is used to set and report the power state of a function as follows: 00b = D0. 01b = D1 (cycle ignored if written with this value). 10b = D2 (cycle ignored if written with this value). 11b = D3.

PMCSR_BSE Bridge Support Extensions – 1 Byte, Offset 0x46, (RO) – This register is not implemented in the 82598; values set to 0x00.

Data Register – 1 Byte, Offset 0x47, (RO) – This optional register is used to report power consumption and heat dissipation. The reported register is controlled by the Data_Select field in the PMCSR; the power scale is reported in the Data_Scale field in the PMCSR. The data of this field is loaded from the EEPROM if power management is enabled in the EEPROM or with a default value of 0x00.

The values for the 82598's functions are as follows:

Function	D0 (Consume/ Dissipate)	D3 (Consume/ Dissipate)	Common	Data_Scale/ Data_Select
	(0x0/0x4)	(0x3/0x7)	(0x8)	
0	EEP PCIe control offset 6	EEP PCIe control offset 6	EEP PCIe control offset 6	01b
1	EEP PCIe control offset 6	EEP PCIe control offset 6	0x00	01b

Note: For other Data_Select values the Data register output is reserved (0b).

3.1.1.14.5 MSI Configuration

This structure is required for PCIe devices. There are no changes to this structure from the initial values of the configuration registers. Defaults are marked in parenthesis.

Color Notation:
- Dotted – Fields that are identical to all functions
- Light-blue – Read-only fields
- Magenta – Hardcoded

Table 3-19. Message Signaled Interrupt Configuration Registers

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x50	Message Control (0x0080)		Next Pointer	Capability ID (0x05)
0x54	Message Address			
0x58	Message Upper Address			
0x5C	Reserved		Message Data	

Capability ID – 1 Byte, Offset 0x50, (RO) – This field equals 0x05, indicating the linked list item as being Message Signaled Interrupt registers.

Next Pointer – 1 Byte, Offset 0x51, (RO) – This field provides an offset to the next item in the capability list. Its value of 0x60 points to the MSI-X capability.

Message Control – 2 Byte, Offset 0x52, (R/W) – These fields are listed in Table 3-20. Note that there is a dedicated register per PCI function to separately enable MSI.

Table 3-20. MSI Message Control Field

Bits	Default	R/W	Description
0	0b	R/W	MSI Enable. 1b = Message Signaled Interrupts. The 82598 generates an MSI for interrupt assertion instead of INTx signaling.
3:1	000b	RO	Multiple Messages Capable. The 82598 indicates a single requested message per function.
6:4	000b	RO	Multiple Message Enable. The 82598 returns 000b to indicate that it supports a single message per function.
7	1b	RO	64-bit Capable. A value of 1b indicates that the 82598 is capable of generating 64-bit message addresses.
15:8	0b	RO	Reserved. Reads as 0b

Message Address Low – 4 Byte, Offset 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 returns 0b regardless of the write operation.

Message Address High – 4 Byte, Offset 0x58, (R/W) – Written by the system to indicate the upper 32 bits of the address to use for the MSI memory write transaction.

Message Data – 2 Byte, Offset 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.

3.1.1.14.6 MSI-X Configuration

The MSI-X capability structure is in Table 3-21. Note that more than one MSI-X capability structure per function is prohibited; however, 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 that begins at 0x10 in the configuration space. A BAR Indicator Register (BIR) indicates which BAR and a Qword-aligned offset indicates where the structure begins relative to the base address associated with the BAR. The BAR can be either 32-bits or 64-bit, but must map to the memory space. A function is permitted to map both structures with the same BAR or map each structure with a different BAR.

The MSI-X table structure (Table 3-25) 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 (Table 3-26) contains the function's pending bits, one per table entry, organized as a packed array of bits within Qwords.

Note: The last Qword will not necessarily be fully populated.

Table 3-21. MSI-X Capability Structure

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x60	Message Control (0x0080)		Next Pointer	Capability ID (0x11)
0x64	Table Offset			Table BIR
0x68	PBA Offset			PBA BIR

Capability ID – 1 Byte, Offset 0x60, (RO) – This field equals 0x11 indicating that the linked list item as being the MSI-X registers.

Next Pointer – 1 Byte, Offset 0x61, (RO) – This field provides an offset to the next capability item in the capability list. Its value of 0xA0 points to PCIe capability.

Message Control – 2 Byte, Offset 0x62, (R/W) – These register fields are listed in Table 3-22.

Table 3-22. MSI-X Message Control Field

Bits	Default	R/W	Description
10:0	0x013	RO	Table Size. System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. The 82598 supports up to 20 different interrupt vectors per function.
13:11	000b	RO	Always returns 000b on a read. A write operation has no effect.
14	0b	R/W	Function Mask. If 1b, all of the vectors associated with the function are masked, regardless of their per-vector *Mask* bit states. If 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	MSI-X Enable. If 1b and the *MSI Enable* bit in the MSI Message Control register is 0b, the function is permitted to use MSI-X to request service and is prohibited from using its INTx# pin. System configuration software sets this bit to enable MSI-X. A device driver is prohibited from writing this bit to mask a function's service request. If 0b, the function is prohibited from using MSI-X to request service.

Table 3-23. MSI-X Table Offset

Bits	Default	R/W	Description
31:3	0x000	RO	Table Offset. Used as an offset from the address contained by one of the function's Base Address registers to point to the base of the MSI-X table. The lower three *Table BIR* bits are masked off (set to 0b) by software to form a 32-bit Qword-aligned offset. Note that this field is read only.
2:0	0x3	RO	Table BIR. Indicates which one of a function's Base Address registers, beginning at 0x10 in the configuration space, is used to map the function's MSI-X table into the memory space. BIR value Base Address register: 0 = 0x10. 1 = 0x14. 2 = 0x18. 3 = 0x1C. 4 = 0x20. 5 = 0x 24. 6 = Reserved. 7 = Reserved. For a 64-bit Base Address register; the table BIR indicates the lower Dword. Hardwired to 0b.

Table 3-24. Table Offset

Bits	Default	R/W	Description
31:3	0x0400	RO	PBA Offset. The offset from the address contained in one of the function Base Address registers; points to the base of the MSI-X PBA. The lower three PBA BIR bits are masked off (set to 0b) by software to form a 32-bit Qword-aligned offset. The field is read only.
2:0	0x3	RO	PBA BIR. Indicates which of a function's Base Address registers, beginning at 0x10 in configuration space, is used to map the function's MSI-X PBA into memory space. PBA BIR value definitions are identical to those for the MSI-X table BIR. This field is read only and set to 0b.

Table 3-25. MSI-X Table Structure

Dword3	Dword2	Dword1	Dword0		
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 0	Base
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: In the 82598, N =16

Table 3-26. MSI-X PBA Structure

Pending bits 0 through 63	Qword0	Base

Note: In the 82598, N = 20. Therefore, only Qword0 is implemented.

Table 3-27. MSI-X Table Structure (Message Address Field)

Bits	Default	Type	Description
31:2	0x00	R/W	Message Address. System-specified message lower address. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the lower portion of the Dword-aligned address (AD[31:02]) for the memory write transaction. This field is read/write.
1:0	0x00	R/W	Message Address. For proper Dword alignment, software must always write zeroes to these two bits; otherwise the result is undefined. The state of these bits after reset must be 0b. These bits are permitted to be read-only or read/write.

Table 3-28. MSI-X Table Structure (Message Upper Address Field)

Bits	Default	Type	Description
31: 0	0x00	R/W	Message Upper Address. System-specified message upper address bits. If this field is zero, Single Address Cycle (SAC) messages are used. If this field is non-zero, Dual Address Cycle (DAC) messages are used. This field is read/write.

Table 3-29. MSI-X Table Structure (Message Data Field)

Bits	Default	Type	Description
31:0	0x00	R/W	Message Data. System-specified message data. For MSI-X messages, the contents of this field from an MSI-X table entry specifies the data driven on AD[31:0] during the memory write transaction's data phase. This field is read/write.

Table 3-30. MSI-X Table Structure (Vector Control Field)

Bits	Default	Type	Description
31:1	0x00	R/W	Reserved. After reset, the state of these bits must be 0b. However, for potential future use, software must preserve the value of these reserved bits when modifying the value of other *Vector Control* bits. If software modifies the value of these reserved bits, the result is undefined.
0	1b	R/W	Mask Bit. When this bit is set, the function is prohibited from sending a message using this MSI-X table entry. However, any other MSI-X table entries programmed with the same vector are still capable of sending an equivalent message unless they are also masked. This bit's state after reset is 1b (entry is masked). This bit is read/write.

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, and 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

3.1.1.14.7 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.

Initialization values of the Configuration registers are marked in parenthesis.

Color Notation:

Dotted – Fields that are identical to all functions

Light-blue – Read-only fields

Magenta – Hardcoded

PCIe Capability Structure – The 82598 implements the PCIe capability structure for endpoint devices as follows:

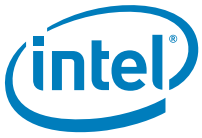
Table 3-31. PCIe Configuration Registers

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0xA0	PCIe Capability Register		Next Pointer	Capability ID
0xA4	Device Capability			
0xA8	Device Status		Device Control	
0xAC	Link Capability			
0xB0	Link Status		Link Control	
0xB4	Reserved			
0xB8	Reserved		Reserved	
0XBC	Reserved		Reserved	
0xC0	Reserved			
X0C4	Device Capability 2			
0xC8	Reserved		Device Control 2	
0xCC	Reserved			
0xD0	Reserved		Link Control 2	
0XD4	Reserved			
0xD8	Reserved		Reserved	

Capability ID – 1 Byte, Offset 0xA0, (RO) – This field equals 0x10 indicating that the linked list item as being the PCIe Capabilities Registers.

Next Pointer – 1 Byte, Offset 0xA1, (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.

PCIe CAP – 2 Byte, Offset 0xA2, (RO) – The PCIe Capabilities register identifies PCIe device type and associated capabilities. This is a read-only register identical to all functions.



Bits	Default	R/W	Description
3:0	0010b	RO	Capability Version. Indicates the PCIe capability structure version. The 82598 supports both version 1 and version 2 as loaded from the *PCIe Capability Version* bit in the EEPROM.
7:4	0000b	RO	Device/Port Type. Indicates the type of PCIe functions. All functions are native PCI functions with a value of 0000b.
8	0b	RO	Slot Implemented. The 82598 does not implement slot options. Therefore, this field is hardwired to 0b.
13:9	00000b	RO	Interrupt Message Number. The 82598 does not implement multiple MSI per function. As a result, this field is hardwired to 0x0.
15:14	00b	RO	Reserved.

Device CAP – 4 Byte, Offset 0xA4, (RO) – This register identifies the PCIe device specific capabilities. It is a read-only register with the same value for the two LAN functions and to all other functions.

Bits	R/W	Default	Description
2:0	RO	001b	Max Payload Size Supported. This field indicates the maximum payload that the 82598 can support for TLPs. It is loaded from the EEPROM with a default value of 256 bytes.
4:3	RO	00b	Reserved.
5	RO	0b	Extended Tag Field Supported. Maximum supported size of the *Tag* field. The 82598 supports a 5-bit *Tag* field for all functions.
8:6	RO	011b	Endpoint L0s Acceptable Latency. This field indicates the acceptable latency that the 82598 can withstand due to the transition from L0s state to the L0 state. All functions share the same value loaded from the EEPROM PCIe Init Configuration 1 bits [8:6]. A value of 011b equals 512 ns.
11:9	RO	110b	Endpoint L1 Acceptable Latency. This field indicates the acceptable latency that the 82598 can withstand due to the transition from L1 state to the L0 state. The 82598 does not support ASPM L1. A value of 110b equals 32 µs-64 µs.
12	RO	0b	Attention Button Present. Hardwired in the 82598 to 0b for all functions.
13	RO	0b	Attention Indicator Present. Hardwired in the 82598 to 0b for all functions.
14	RO	0b	Power Indicator Present. Hardwired in the 82598 to 0b for all functions.
15	RO	1b	Role Based Error Reporting. Hardwired in the 82598 to 1b for all functions.
17:16	RO	00b	Reserved, should be set to 00b.
25:18	RO	0x00	Slot Power Limit Value. Used in upstream ports only. Hardwired in the 82598 to 0x00 for all functions.
27:26	RO	00b	Slot Power Limit Scale. Used in upstream ports only. Hardwired in the 82598 to 0b for all functions.
31:28	RO	0000b	Reserved.

Device Control – 2 Byte, Offset 0xA8, (RW) – This register controls the PCIe specific parameters. Note that there is a dedicated register per each function.

Bits	R/W	Default	Description
0	R/W	0b	Correctable Error Reporting Enable. Enable error report.
1	R/W	0b	Non-Fatal Error Reporting Enable. Enable error report.
2	R/W	0b	Fatal Error Reporting Enable. Enable error report.
3	R/W	0b	Unsupported Request Reporting Enable. Enable error report.
4	R/W	1b	Enable Relaxed Ordering. If this bit is set, the 82598 is permitted to set the *Relaxed Ordering* bit in the attribute field of write transactions that do not need strong ordering. Refer to the CTRL_EXT register bit *RO_DIS* for more details.
7:5	R/W	000b (128 bytes)	Max Payload Size. This field sets the maximum TLP payload size for the 82598 functions. As a receiver, the 82598 must handle TLPs as large as the set value. As a transmitter, the 82598 must not generate TLPs exceeding the set value. The *Max Payload Size* field supported in the Device Capabilities register indicates permissible values that can be programmed.
9:8	R/W	00b	Reserved, should be set to 00b.
10	R/W	0b	Auxiliary Power PM Enable. When set, enables the 82598 to draw AUX power independent of PME AUX power. the 82598 is a multi-function device, therefore allowed to draw AUX power if at least one of the functions has this bit set.
11	R/W	1b	Enable No Snoop. Snoop is gated by *Non-Snoop* bits in the GCR register in the CSR space.
14:12	R/W	010b	Max Read Request Size. This field sets maximum read request size for the 82598 as a requester. 000b = 128 bytes. 001b = 256 bytes. 010b = 512 bytes. 011b = 1 kB. 100b = 2 kB. 101b = Reserved. 110b = Reserved. 111b = Reserved.
15	RO	0b	Reserved. **Note:** The 82598 does not issue read requests greater than 256 bytes.

Device Status – 2 Byte, Offset 0xAA, (RO) – This register provides information about PCIe device specific parameters. Note that there is a dedicated register per each function.

Bits	R/W	Default	Description
0	RW1C	0b	Correctable Detected. Indicates status of correctable error detection.
1	RW1C	0b	Non-Fatal Error Detected. Indicates status of non-fatal error detection.
2	RW1C	0b	Fatal Error Detected. Indicates status of fatal error detection.
3	RW1C	0b	Unsupported Request Detected. Indicates that the 82598 received an unsupported request. This field is identical in all functions. the 82598 can't distinguish which function causes the error.

4	RO	0b	Aux Power Detected. If Aux Power is detected, this field is set to 1b. It is a strapping signal from the periphery and is identical for all functions. Resets on LAN Power Good, internal power on reset, and PE_RST_N only.
5	RO	0b	Transaction Pending. Indicates whether the 82598 has ANY transactions pending. (Transactions include completions for any outstanding non-posted request for all used traffic classes).
15:6	RO	0x00	Reserved.

Link CAP – 4 Byte, Offset 0xAC, (RO) – This register identifies PCIe link-specific capabilities. This is a read-only register identical to all functions.

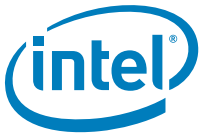
Bits	R/W	Default	Description
3:0	RO	0001b	Supported Link Speeds. This field indicates the supported Link speed(s) of the associated link port. Defined encodings are: 0001b = 2.5 Gb/s Link speed supported. 0010b = 5 Gb/s and 2.5 Gb/s Link speeds supported.
9:4	RO	0x08	Max Link Width. Indicates the maximum link width. The 82598 supports a x1, x2, x4 and x8-link width. This field is loaded from the EEPROM PCIe init configuration 3 Word 0x1A with a default value of eight lanes. Defined encoding: 000000b = Reserved 000001b = x1 000010b = x2 000100b = x4 001000b = x8
11:10	RO	01b	Active State Link PM Support. Indicates the level of the active state of power management supported in the 82598. Defined encodings are: 00b = Reserved 01b = L0s entry supported 10b = Reserved 11b = L0s and L1 supported This field is loaded from the EEPROM PCIe init configuration 3 Word 0x1A.
Bits	**R/W**	**Default**	**Description**
14:12	RO	110b[1] (2 “s – 4 “s)	L0s Exit Latency. Indicates the exit latency from L0s to L0 state. 000b = Less than 64 ns 001b = 64 ns – 128 ns 010b – 128 ns – 256 ns 011b – 256 ns – 512 ns 100b = 512 ns Π l “s 101b = 1 “s – 2 “s 110b = 2 “s – 4 “s 111b = Reserved

17:15	RO	111b	L1 Exit Latency. Indicates the exit latency from L1 to L0 state. The 82598 does not support ASPM L1. 000b = Less than 1 “s 001b = 1 “s – 2 “s 010b = 2 “s – 4 “s 011b = 4 “s – 8 “s 100b = 8 “s – 16 “s 101b = 16 “s – 32 “s 110b = 32 “s – 64 “s 111b = L1 transition not supported.
18	RO	0b	Clock Power Management.
19	RO	0b	Surprise Down Error Reporting Capable.
20	RO	0b	Data Link Layer Link Active Reporting Capable.
21	RO	0b	Link Bandwidth Notification Capability.
23:22	RO	00b	Reserved.
31:24	HwInit	0x0	Port Number. The PCIe port number for the given PCIe link. This field is set in the link training phase.

1. Loaded from the EEPROM.

Link Control – 2 Byte, Offset 0xB0, (RO) – This register controls PCIe Link specific parameters. There is a dedicated register per each function.

Bits	R/W	Default	Description
1:0	R/W	00b	Active State Link PM Control. This field controls the active state PM supported on the link. Link PM functionality is determined by the lowest common denominator of all functions. Bit 0 of this field is loaded from PCIe init configuration 1, offset 1, bit 15 (L0s Enable). Defined encodings are: 00b = PM disabled. 01b = L0s entry supported. 10b = Reserved. 11b = L0s and L1 supported.
2	RO	0b	Reserved.
3	R/W	0b	Read Completion Boundary.
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	R/W	0b	Common Clock Configuration. When set, indicates that the 82598 and the component at the other end of the link are operating with a common reference clock. A value of 0b indicates that they are operating with an asynchronous clock. This parameter affects the L0s exit latencies.
7	R/W	0b	Extended Sync. When set, this bit forces an extended Tx of the FTS ordered set in FTS and an extra TS1 at the exit from L0s prior to entering L0.
8	RO	0b	Reserved.
9	RO	Hardwired to 0b	Hardware Autonomous Width Disable. When set to 1b, this bit disables hardware from changing the link width for reasons other than attempting to correct an unreliable link operation by reducing link width.



11:10	RO	00b	Reserved. Read only as 00b.
15:12	RO	0000b	Reserved.

Link Status – 2 Byte, Offset 0xB2, (RO) – This register provides information about PCIe Link specific parameters. This is a read only register identical to all functions.

Bits	R/W	Default	Description
3:0	RO	0001b	Current Link Speed. This field indicates the negotiated link speed of the given PCIe link. Defined encodings are: 0001b = 2.5 Gb/s PCIe link. 0010b = 5 Gb/s PCIe link. All other encodings are reserved.
9:4	RO	000001b	Negotiated Link Width. Indicates the negotiated width of the link. Relevant encodings for the 82598 are: 000001b = x1. 000010b = X2. 000100b = x4. 001000b = x8.
10	RO	0b	Link Training Error. Indicates that a link training error has occurred.
11	RO	0b	Link Training. Indicates that link training is in progress.
12	HwInit	1b	Slot Clock Configuration. When set, indicates that the 82598 uses the physical reference clock that the platform provides at the connector. This bit must be cleared if the 82598 uses an independent clock. The *Slot Clock Configuration* bit is loaded from the *Slot_Clock_Cfg* EEPROM bit.
14:13	RO	00b	Reserved. Read only as 00b.
15	RO	0b	Reserved.

The following registers are supported only if the capability version is two and above.

Device CAP 2 – 4 Byte, Offset 0xC4, (RO) – This register identifies the PCIe device-specific capabilities. It is a read-only register with the same value for both LAN functions.

Bits	R/W	Default	Description
3:0	RO	1111b	Completion Timeout Ranges Supported. This field indicates the 82598's support for the optional completion timeout programmability mechanism. Four time value ranges are defined: • Range A: 50 µs to 10 ms. • Range B: 10 ms to 250 ms. • Range C: 250 ms to 4 s. • Range D: 4 s to 64 s. Bits are set according to the following values to show the timeout value ranges that the 82598 supports. • 0000b = Completion timeout programming not supported. the 82598 must implement a timeout value in the range of 50 µs to 50 ms. • 0001b = Range A. • 0010b = Range B. • 0011b = Ranges A and B. • 0110b = Ranges B and C. • 0111b = Ranges A, B and C. • 1110b = Ranges B, C and D. • 1111b = Ranges A, B, C and D. • All other values are reserved.
Bits	**R/W**	**Default**	**Description**
4	RO	1b	Completion Timeout Disable Supported.
31:5	RO	0b	Reserved.

Device Control 2 – 2 Byte, Offset 0xC8, (RW) – This register controls the PCIe specific parameters. Note that there is a dedicated register per each function.

Bits	R/W	Default	Description
3:0	RW	0b	Completion Timeout Value. For devices that support completion timeout programmability, this field enables system software to modify the completion timeout value. Defined encodings: • 0000b = Default range: 50 µs to 50 ms. **Note:** It is strongly recommended that the completion timeout mechanism not expire in less than 10 ms. Values available if Range A (50 µs to 10 ms) programmability range is supported: • 0001b = 50 µs to 100 µs. • 0010b = 1 ms to 10 ms. Values available if Range B (10 ms to 250 ms) programmability range is supported: • 0101b = 16 ms to 55 ms. • 0110b = 65 ms to 210 ms. Values available if Range C (250 ms to 4 s) programmability range is supported: • 1001b = 260 ms to 900 ms. • 1010b = 1 s to 3.5 s. Values available if the Range D (4 s to 64 s) programmability range is supported: • 1101b = 4 s to 13 s. • 1110b = 17 s to 64 s. Values not defined are reserved. Software is permitted to change the value of this field at any time. For requests already pending when the completion timeout value is changed, hardware is permitted to use either the new or the old value for the outstanding requests and is permitted to base the start time for each request either on when this value was changed or on when each request was issued.

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.
15:5	RO	0b	Reserved.

Link Control 2 – 2 Byte, Offset 0xD0, (RW)

Bits	R/W	Default	Description
3:0	RW	See description	Target Link Speed. This field is used to set the target compliance mode speed when software is using the *Enter Compliance* bit to force a link into compliance mode. Defined encodings are: 0001b = 2.5 Gb/s target link speed. 0010b = 5 Gb/s target link speed. All other encodings are reserved. If a value is written to this field that does not correspond to a speed included in the *Supported Link Speeds* field, the result is undefined. The default value of this field is the highest link speed supported by the 82598 (as reported in the *Supported Link Speeds* field of the Link Capabilities register) unless the corresponding platform/form factor requires a different default value.
4	RW	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	RW	Hardwired to 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. If the 82598 does not implement the associated mechanism it is permitted to hardwire this bit to 0b.

3.1.1.14.8 PCIe Extended Configuration Space

PCIe configuration space is located in a flat memory-mapped address space. PCIe extends the configuration space beyond the 256 bytes available for PCI to 4096 bytes. The 82598 decodes an additional four bits (bits 27:24) to provide the additional configuration space as shown. PCIe reserves the remaining four bits (bits 31:28) for future expansion of the configuration space beyond 4096 bytes.

The configuration address for a PCIe device is computed using a PCI-compatible bus, device, and function numbers as follows:

31 28	27 20	19 15	14 12	11 2	1 0
0000b	Bus #	Device #	Funct #	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 82598 supports the following PCIe extended capabilities:

- Advanced Error Reporting Capability – offset 0x100
- Serial Number Capability – offset 0x140

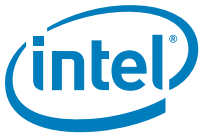
3.1.1.14.8.1 Advanced Error Reporting Capability

The PCIe advanced error reporting capability is an optional extended capability to support advanced error reporting. The tables that follow list the PCIe advanced error reporting extended capability structure for PCIe devices.

Register Offset	Field	Description
0x00	PCIe CAP ID	PCIe Extended Capability ID.
0x04	Uncorrectable Error Status	Reports error status of individual uncorrectable error sources on a PCIe device.
0x08	Uncorrectable Error Mask	Controls reporting of individual uncorrectable errors by device to the host bridge via a PCIe error message.
0x0C	Uncorrectable Error Severity	Controls whether an individual uncorrectable error is reported as a fatal error.
0x10	Correctable Error Status	Reports error status of individual correctable error sources on a PCIe device.
0x14	Correctable Error Mask	Controls reporting of individual correctable errors by device to the host bridge via a PCIe error message.
0x18	Advanced Error Capabilities and Control	Identifies the bit position of the first uncorrectable error reported in the Uncorrectable Error Status register.
0x1C:0x28	Header Log	Captures the header for the transaction that generated an error.

PCIe CAP ID

Bit Location	Attribute	Default Value	Description
15:0	RO	0x0001	Extended Capability ID. PCIe extended capability ID indicating advanced error reporting capability.
19:16	RO	0x1	Version Number. PCIe advanced error reporting extended capability version number.
31:20	RO	0x0140	Next Capability Pointer. Next PCIe extended capability pointer.



Uncorrectable Error Status

The Uncorrectable Error Status register reports error status of individual uncorrectable error sources on a PCIe device. An individual error status bit that is set to 1b indicates that a particular error occurred; software can clear an error status by writing a 1b to the respective bit.

Bit Location	Attribute	Default Value	Description
3:0	RO	0b	Reserved.
4	R/W1CS	0b	Data Link Protocol Error Status.
11:5	RO	0b	Reserved.
12	R/W1CS	0b	Poisoned TLP Status.
13	R/W1CS	0b	Flow Control Protocol Error Status.
14	R/W1CS	0b	Completion Timeout Status.
15	R/W1CS	0b	Completer Abort Status.
16	R/W1CS	0b	Unexpected Completion Status.
17	R/W1CS	0b	Receiver Overflow Status.
18	R/W1CS	0b	Malformed TLP Status.
19	RO	0b	Reserved.
20	R/W1CS	0b	Unsupported Request Error Status.
31:21	Reserved	0b	Reserved.

Uncorrectable Error Mask

The Uncorrectable Error Mask register controls reporting of individual uncorrectable errors by device to the host bridge via a PCIe error message. A masked error (respective bit set in mask register) is not reported to the host bridge by an individual device. Note that there is a mask bit per bit of the Uncorrectable Error Status register.

Bit Location	Attribute	Default Value	Description
3:0	RO	0b	Reserved.
4	RWS	0b	Data Link Protocol Error Mask.
11:5	RO	0b	Reserved.
12	RWS	0b	Poisoned TLP Mask.
13	RWS	0b	Flow Control Protocol Error Mask.
14	RWS	0b	Completion Timeout Mask.
15	RWS	0b	Completer Abort Mask.
16	RWS	0b	Unexpected Completion Mask.

Bit Location	Attribute	Default Value	Description
17	RWS	0b	Receiver Overflow Mask.
18	RWS	0b	Malformed TLP Mask.
19	RO	0b	Reserved.
20	RWS	0b	Unsupported Request Error Mask.
31:21	Reserved	0b	Reserved.

Uncorrectable Error Severity

The Uncorrectable Error Severity register controls whether an individual uncorrectable error is reported as a fatal error. An uncorrectable error is reported as fatal when the corresponding error bit in the severity register is set. If the bit is cleared, the corresponding error is considered fatal. If the bit is set, the corresponding error is considered non-fatal.

Bit Location	Attribute	Default Value	Description
3:0	RO	0b	Reserved.
4	RWS	1b	Data Link Protocol Error Severity.
11:5	RO	0b	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	RO	0b	Reserved.
20	RWS	1b	Unsupported Request Error Severity.
31:21	Reserved	0b	Reserved.

Correctable Error Status

The Correctable Error Status register reports error status of individual correctable error sources on a PCIe device. When an individual error status bit is set to 1b it indicates that a particular error occurred; software can clear an error status by writing a 1b to the respective bit.

Bit Location	Attribute	Default Value	Description
0	R/W1CS	0b	Receiver Error Status.
5:1	RO	0b	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	0b	Reserved.
12	R/W1CS	0b	Replay Timer Timeout Status.
13	R/W1CS	0b	Advisory Non Fatal Error Status.
15:14	RO	0b	Reserved.

Correctable Error Mask

The Correctable Error Mask register controls reporting of individual correctable errors by device to the host bridge via a PCIe error message. A masked error (respective bit set in mask register) is not reported to the host bridge by an individual device. There is a mask bit per bit in the Correctable Error Status register.

Bit Location	Attribute	Default Value	Description
0	RWS	0b	Receiver Error Mask.
5:1	RO	0b	Reserved.
6	RWS	0b	Bad TLP Mask.
7	RWS	0b	Bad DLLP Mask.
8	RWS	0b	REPLAY_NUM Rollover Mask.
11:9	RO	0b	Reserved.
12	RWS	0b	Replay Timer Timeout Mask.
13	RWS	1b	Advisory Non Fatal Error Mask.
15:14	RO	0b	Reserved.

Advanced Error Capabilities and Control

The First Error Pointer is a read-only register that identifies the bit position of the first uncorrectable error reported in the Uncorrectable Error Status register.

Bit Location	Attribute	Default Value	Description
4:0	RO	0b	Vector pointing to the first recorded error in the Uncorrectable Error Status register.

Header Log

The header log register captures the header for the transaction that generated an error. This register is 16 bytes.

Bit Location	Attribute	Default Value	Description
127:0	RO	0b	Header of the packet in error (TLP or DLLP).

3.1.1.14.8.2 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.

All multi-function devices that implement this capability must implement it for function 0; other functions that implement this capability must return the same device serial number value as that reported by function 0.

Table 3-32. PCIe Device Serial Number Capability Structure

31:0	Address
PCIe Enhanced Capability Header.	0x00
Serial Number Register (Lower Dword).	0x04
Serial Number Register (Upper Dword).	0x08

Device Serial Number Enhanced Capability Header (Offset 0x00)

Table 3-33 lists the allocation of register fields in the device serial number enhanced capability header. It provides the respective bit definitions. Section 3.1.1.14.8 for a description of the PCIe enhanced capability header. The extended capability ID for the device serial number capability is 0x0003.

31:20	19:16	15:0
Next Capability Offset	Capability Version	PCIe Extended Capability ID

Table 3-33. Device Serial Number Enhanced Capability Header

Bit(s) Location	Attributes	Description
15:0	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	RO	Capability Version. This field is a PCI-SIG defined version number that indicates the version of the capability structure present. **Note:** Must be set to 0x1 for this version of the specification.
31:20	RO	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. For extended capabilities implemented in the device configuration space, this offset is relative to the beginning of PCI-compatible configuration space and must always be either 0x000 (for terminating list of capabilities) or greater than 0x0FF.

Serial Number Register (Offset 0x04)

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit Extended Unique Identifier (EUI-64*). Figure 3-7 details the allocation of register fields in the Serial Number register. The table that follows Figure 3-7 lists the respective bit definitions.

31:0
Serial Number Register (Lower Dword).
Serial Number Register (Upper Dword).
63:32

Figure 3-7. Serial Number Register Contents

Bit(s) Location	Attributes	Description
63:0	RO	PCIe Device Serial Number. This field contains the IEEE defined 64-bit EUI-64*. This identifier includes a 24-bit company ID value assigned by IEEE registration authority and a 40-bit extension identifier assigned by the manufacturer.

The serial number uses the MAC address according to the following definition:

Field	Company ID			Extension Identifier				
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7
Most Significant Byte						Least Significant Byte		
Most Significant Bit						Least Significant Bit		

The serial number can be constructed from the 48-bit MAC address in the following form:

Field	Company ID			MAC Label		Extension identifier		
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7
Most Significant Bytes						Least Significant Byte		
Most Significant Bit						Least Significant Bit		

In this case, the MAC label is 0xFFFF.

For example, assume that the company ID is (Intel) 00-A0-C9 and the extension identifier is 23-45-67. In this case, the 64-bit serial number is:

Field	Company ID			MAC Label		Extension Identifier		
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7
	00	A0	C9	FF	FF	23	45	67
Most Significant Byte						Least Significant Byte		
Most Significant Bit						Least Significant Bit		

The MAC address is the function 0 MAC address that is loaded from EEPROM into the RAL and RAH registers.

Note: The official document that defines EUI-64* is: http://standards.ieee.org/regauth/oui/tutorials/EUI64.html

Note: For EEPROM-less configuration, the serial number capability is not supported.

3.1.2 Manageability Interfaces (SMBus/NC-SI)

The 82598 supports pass-through manageability through an on-board BMC. The BMC can be either a stand-alone device or integrated into an Input/Output Hub (IOH). The link between the 82598 and the BMC is NC-SI, SMBus, or a combination of both (see Table 3-34).

Table 3-34 lists the different options for the 82598 manageability links.

Table 3-34. 82598 Manageability Links

Configuration	Interfaces	Pass Through	Configuration
BMC legacy	SMBus	X	X
DMTF standard	NC-SI	X	X
NC-SI back-up mode	NC-SI	X	-
	SMBus	-	X

The 82598 supports two interfaces to an external BMC:

- SMBus
- NC-SI

Note: Since the manageability sideband throughput is lower than the network link throughput, the 82598 allocates an 8 kB internal buffer for incoming network packets prior to being sent over the sideband interface.

3.1.2.1 SMBus Pass-Through Interface

SMBus is the system management bus defined by Intel® Corporation in 1995. It is used in personal computers and servers for low-speed system management communications. The SMBus interface is one of two pass-through interfaces available in the 82598.

3.1.2.1.1 General

The SMBus sideband interface includes the standard SMBus commands used for assigning a slave address and gathering device information for the pass-through interface.

3.1.2.1.2 Pass-Through Capabilities

When operating in SMBus mode, in addition to exposing a communication channel to the LAN for the BMC, the 82598 provides the following manageability services to the BMC:

- ARP handling - The 82598 can be programmed to auto-ARP replying for ARP request packets to reduce the traffic over the BMC interconnect.
- Teaming and fail-over - The 82598 can be configured to one of several teaming and fail-over configurations:
  - — No-teaming - The 82598 dual LAN ports act independently of each other and no fail-over is provided by the 82598. The BMC is responsible for teaming and failover.
  - — Teaming - The 82598 is configured to provide fail-over capabilities, such that manageability traffic is routed to an active port if any of the ports fail. Several modes of operation are provided.

Note: These services are not available in NC-SI mode.

For more information on the SMBus and NC-SI manageability interfaces, refer to the *Intel® 82598 10 GbE Controller System Manageability Interface* application note.This document is available from your Intel representative.

3.1.2.2 NC-SI

The NC-SI interface in the 82598 is a connection to an external BMC. It operates in one of two modes:

- NC-SI-SMBus mode – In conjunction with an SMBus interface, where pass-through traffic passes through NC-SI and configuration traffic passes through SMBus.
- NC-SI mode – As a single interface with an external BMC, where all traffic between the 82598 and the BMC flows through the interface.

3.1.2.2.1 Interface Specification

The 82598 NC-SI interface meets the NC-SI Specification, Rev. 1.2 as a PHY-side device.

The following NC-SI capabilities are not supported by the 82598:

- Collision Detection: The interface supports only full-duplex operation
- MDIO: MDIO/MDC management traffic is not passed on the interface
- Magic Packets: Magic packets are not detected at the 82598 receive end
- Flow Control: The 82598 supports receiving flow control on this interface but no transmission

3.1.2.2.2 Electrical Characteristics

The 82598 complies with the electrical characteristics defined in the DMTF NC-SI specification. However, the 82598 pads are not 5V tolerant and require that signals conform to 3.3V signaling.

NC-SI behavior is configured by the 82598 at power up:

- The output driver strength for the NC-SI output signals (NC-SI_DV and NC-SI_RX) is configured by the EEPROM *NC-SI Data Pad Drive Strength* bit; word 15h, bit 14 (default = 0b).
- The NC-SI topology is loaded from the EEPROM (point-to-point or multi-drop – default is point-to-point)

The 82598 dynamically drives its NC-SI output signals (NC-SI_DV and NC-SI_RX) as required by the sideband protocol:

- At power up, the 82598 floats the NC-SI outputs.
- If the 82598 operates in point-to-point mode, then it starts driving the NC-SI outputs at some time following power up.
- If the 82598 operates in a multi-drop mode, it drives the NC-SI outputs as configured by the BMC.

3.1.2.2.3 NC-SI Transactions

The NC-SI link supports both pass through traffic between the BMC and the 82598 LAN functions as well as configuration traffic between the BMC and the 82598 internal units.

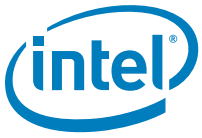
3.1.2.2.3.1 NC-SI-SMBus Mode

NC-SI serves in this mode to transfer pass-through traffic between the BMC and the LAN ports. Packet structure follows the RMI specification as defined in the NC-SI specification. The following limitations apply:

- VLAN traffic (if exists) is carried by the packet in its designated area. If VLAN strip is enabled in an 82598 LAN port, then the VLAN tag must still exist in a VLAN-enabled packet when it is sent over NC-SI to the BMC.
- The *FCS* field must be present on any NC-SI packet sent to the BMC. If packet CRC strip is enabled in the 82598 LAN port, the *FCS* field must still be there when a packet is sent over NC-SI to the BMC.
- Flow-control – The 82598 does not initiate flow control over NC-SI (does not send PAUSE packets). However, the 82598 responds to flow control packets received over NC-SI and meets the flow-control protocol.

3.1.2.2.3.2 NC-SI Mode

This mode is compatible with the pre-OS sideband DMTF standard.



3.1.3 Non-Volatile Memory (EEPROM/Flash)

This section describes the EEPROM and Flash interfaces supported by 82598.

3.1.3.1 EEPROM

The 82598 uses an EEPROM device to store product configuration information. The EEPROM is divided into three general regions:

- **Hardware Accessed —** Loaded by the 82598 hardware after power-up, PCI reset de-assertion, a D3 to D0 transition, or a software reset.
- **Firmware Area** — Includes structures used by the firmware for management configuration in its different modes. Refer to the *Intel® 82598 10 GbE Controller System Manageability Interface* application note for configuration values
- **Software Accessed —** Used by software only. These registers are listed in this document for convenience and are only for software and are ignored by the 82598.

The EEPROM interface supports Serial Peripheral Interface (SPI) and expects the EEPROM to be capable of 2 MHz operation.

The 82598 is compatible with many sizes of 4-wire serial EEPROM devices. A 4096-bit serial SPI-compatible EEPROM can be used. All EEPROMs are accessed in 16-bit words although the EEPROM is designed to also accept 8-bit data accesses.

The 82598 automatically determines the address size to be used with the SPI EEPROM it is connected to and sets the *EEPROM Size* field of the EEPROM/Flash Control (EEC) and Data Register (EEC.EE_ADDR_SIZE; bit 10). Software uses this size to determine the EEPROM access method. The exact size of the EEPROM is stored within one of the EEPROM words.

The different EEPROM sizes have two different numbers of address bits (8 bits or 16 bits). As a result, they must be accessed with a slightly different serial protocol. Software must be aware of this if it accesses the EEPROM using direct access.

3.1.3.1.1 Software Accesses

The 82598 provides two different methods for software access to the EEPROM. It can either use the built-in controller to read the EEPROM or access the EEPROM directly using the EEPROM's 4-wire interface.

Software can use the EEPROM Read register (EERD) to cause the 82598 to read a word from the EEPROM that the software can then use. To do this, software writes the address to read into the *Read Address* field (EERD.ADDR; bits 15:2) and simultaneously writes a 1b to the *Start Read* bit (EERD.START; bit 0). The 82598 then reads the word from the EEPROM, sets the *Read Done* bit (EERD.DONE; bit 1), and puts the data in the *Read Data* field (EERD.DATA; bits 31:16). Software can poll the EEPROM Read register until it sees the *Read Done* bit set, then use the data from the *Read Data* field. Any words read this way are not written to the 82598's internal registers.

Software can also directly access the EEPROM's 4-wire interface through the EEPROM/Flash Control register (EEC). It can use this for reads, writes, or other EEPROM operations.

To directly access the EEPROM, software should follow these steps:

1. Write a 1b to the *EEPROM Request* bit (EEC.EE_REQ; bit 6).
2. Read the *EEPROM Grant* bit (EEC.EE_GNT; bit 7) until it becomes 1b. It remains 0b as long as the hardware is accessing the EEPROM.
3. Write or read the EEPROM using the direct access to the 4-wire interface as defined in the EEPROM/Flash Control & Data register (EEC). The exact protocol used depends on the EEPROM placed on the board and can be found in the appropriate datasheet.

4. Write a 0b to the *EEPROM Request* bit (EEC.EE_REQ; bit 6).

Each time the EEPROM is not valid (blank EEPROM or wrong signature), software should use the direct access to the EEPROM through the EEC register.

3.1.3.1.2 Signature Field

The only way the 82598 can discover whether an EEPROM is present is by trying to read the EEPROM. The 82598 first reads the EEPROM Control Word at address 0x0. The 82598 checks the signature value for bits 7 and 6. If bit 7 is 0b and bit 6 is 1b, it considers the EEPROM to be present and valid and reads additional EEPROM words and programs its internal registers based on the values read. Otherwise, it ignores the values it read from that location and does not read any other words.

3.1.3.1.3 Protected EEPROM Space

The 82598 provides to the host a mechanism for a hidden area in the EEPROM. The hidden area cannot be accessed via the EEPROM registers in the CSR space. It can be accessed only by the Manageability (MNG) subsystem. For more information on the MNG subsystem, refer to the *Intel® 82598 10 GbE Controller System Manageability Interface* application note.

After the EEPROM is configured to be protected, changing bits that are protected require specific manageability instructions with authentication mechanism. This mechanism is defined in the *Intel® 82598 10 GbE Controller System Manageability Interface* application note.

3.1.3.1.3.1 Initial EEPROM Programming

In most applications, initial EEPROM programming is done directly on the EEPROM pins. Nevertheless, it is desirable to enable existing software utilities (accessing the EEPROM via the host interface) to initially program the whole EEPROM without breaking the protection mechanism. Following a power-up sequence, the 82598 reads the hardware initialization words in the EEPROM. If the signature in word 0x0 does not equal 01b the EEPROM is assumed as non-programmed. There are two effects for non-valid signature:

- The 82598 stops reading EEPROM data and sets the relevant registers to default values.
- The 82598 enables access to any location in the EEPROM via the EEPROM CSR registers.

3.1.3.1.3.2 EEPROM Protected Areas

The 82598 defines two protected areas in the EEPROM. The first area is words 0x00-0x0F these words hold the basic configuration and the pointers to all other configuration sections. The second area is a programmable size area located at the end of the EEPROM and targeted at protecting the appropriate sections that should be blocked for changes.

3.1.3.1.3.3 Activating the Protection Mechanism

Following an 82598 initialization, it reads the Init Control word from the EEPROM. It then turns on the protection mechanism if word 0x0h [7:6] contains a valid signature (equals 01b) and bit 4 in word 0x0 is set to 1b (enable protection). Once the protection mechanism is turned on, word 0x0 becomes write-protected and the area that is defined by word 0x0 becomes hidden (for example, read/write protected).

Although possible by configuration, it is prohibited that the software section in the EEPROM be included as part of the EEPROM protected area.

3.1.3.1.3.4 Non Permitted Accesses to Protected Areas in the EEPROM

This section refers to EEPROM accesses via the EEC (bit banging) or EERD (parallel read access) registers. Following a write access to the write protected areas in the EEPROM, the hardware responds properly on the PCIe bus, but does not initiate any access to the EEPROM. Following a read access to the hidden area in the EEPROM (as defined by word 0x0), the hardware does not access the EEPROM and returns meaningless data to the host.

Using bit banging, the SPI EEPROM can be accessed in a burst mode. For example, providing an opcode address and then reading or writing data for multiple bytes. The hardware inhibits an attempt to access the protected EEPROM locations even in burst accesses.

Software should not access the EEPROM in a Burst Write mode starting in a non protected area and continue to a protected one. In such a case, it is not guaranteed that the write access to any area ever takes place.

3.1.3.1.4 EEPROM Recovery

The EEPROM contains fields that if programmed incorrectly might affect the functionality of 82598. The impact can range from incorrectly setting a function like LED programming, disabling an entire feature like no manageability or link disconnection, to the inability to access the 82598 via the regular PCIe interface.

The 82598 implements a mechanism that enables a recovery from a faulty EEPROM no matter what the impact is by using an SMBus message that instructs the firmware to invalidate the EEPROM.

This mechanism uses an SMBus message that the firmware is able to receive in all modes, no matter what the content of the EEPROM is (even in diagnostic mode). After receiving this kind of message, the firmware clears the signature of the EEPROM in word 0x0 bit 7/6 to 00b. Afterwards, the BIOS/ operating system initiates a reset to force an EEPROM auto-load process that fails and enables access to the 82598.

Firmware is programmed to receive such a command only from a PCIe reset until one of the functions changes it status from D0u to D0a. Once one of the functions switches to D0a, it can be safely assumed that the 82598 is accessible to the host and there is no more need for this function. This reduces the possibility of malicious software to use this command as a back door and limits the time the firmware must be active in non-manageability mode.

The command is sent on a fixed SMBus address of 0xC8. The format of the command is SMBus Write Data Byte as follows:

Function	Command	Data Byte
Release EEPROM[1]	0xC7	0xB6

1. This solution requires a controllable SMBus connection to the 82598. If more than one 82598 is in a state to accept this solution, then all the 82598s connected to the same SMBus accepts the command. The 82598s in D0u release the EEPROM.

After receiving a release EEPROM command, firmware should keep its current state. It is the responsibility of the programmer updating the EEPROM to send a firmware reset, if required, after the full EEPROM update process completes.

Data byte 0xB6 is the LSB of the 82598's default device ID.

An additional command is introduced to enable the EEPROM write directly from the SMBus interface to enable the EEPROM modification (writing from the SMBus to any MAC CSR register). The same rules as for the Release EEPROM command that determine when the firmware accepts this command apply to this command as well.

The Command is sent on a fixed SMBus address of 0xC8. The format of the command is SMBus Block Write is as follow:

Function	Cmd	Byte Count	Data 1	Data 2	Data 3	Data 4	...	Data 7
EEPROM Write	0xC8	7	Config address 2	Config address 1	Config address 0	Config data MSB	...	Config data LSB

The MSB in configuration address 2 indicates which port is the target of the access (0 or 1).

The 82598 always enables the manageability block after power up. The manageability clock is stopped if the manageability function is disabled in the EEPROM and one of the functions had transitioned to D0a; otherwise, the manageability block gets the clock and is able to wait for the new command.

This command enables writing to any MAC CSR register as part of the EEPROM recovery process. This command can also be used to write to the EEPROM and update different sections in it.

3.1.3.2 Flash

The 82598 provides an interface to an external serial Flash/ROM memory device. This Flash/ROM device can be mapped into memory and/or I/O address space for each LAN device through the use of Base Address Registers (BARs). The EEPROM bit associated with each LAN device selectively disables/ enables whether the Flash can be mapped for each LAN device by controlling the BAR register advertisement and write ability.

3.1.3.2.1 Flash Interface Operation

The 82598 provides two different methods for software access to the Flash.

Using legacy Flash transactions, the Flash is read from, or written to, each time the host processor performs a read or a write operation to a memory location that is within the Flash address mapping or at boot via accesses in the space indicated by the Expansion ROM Base Address register. All accesses to the Flash require the appropriate command sequence for the 82598 used. Refer to the specific Flash data sheet for more details on reading from or writing to Flash.

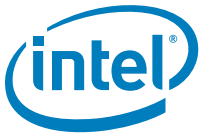
Accesses to the Flash are based on a direct decode of processor accesses to a memory window defined in either:

- The 82598's Flash Base Address register (PCIe Control register at offset 0x14 or 0x18).
- A certain address range of the IOADDR register defined by the IO Base Address register (PCIe Control register at offset 0x18 or 0x20).
- The Expansion ROM Base Address register (PCIe Control register at offset 0x30).

The 82598 controls accesses to the Flash when it decodes a valid access.

Note: Flash read accesses must always be assembled by the 82598 each time the access is greater than a byte-wide access. The component byte reads or writes to the Flash take on the order of 2 μs; it continues to issue retry accesses during this time. The 82598 supports only byte writes to the Flash.

Another way for software to access the Flash is directly using the Flash's 4-wire interface through the Flash Access register (FLA). It can use this for reads, writes, or other Flash operations (accessing the Flash status register, erase, etc.).



To directly access the Flash, software needs to:

- Write a 1b to the Flash Request bit (FLA.FL_REQ)
- Read the Flash Grant bit (FLA.FL_GNT) until it = 1b. It remains 0b as long as there are other accesses to the Flash.
- Write or read the Flash using the direct access to the 4-wire interface as defined in the Flash Access register (FLA). The exact protocol used depends on the Flash placed on the board and can be found in the appropriate Flash datasheet.
- Write a 0b to the *Flash Request* bit (FLA.FL_REQ).

3.1.3.2.2 Flash Write Control

The Flash is write controlled by the FWE bits in the EEPROM/Flash Control and Data register (EEC.FWE). Note that attempts to write to the Flash device when writes are disabled (FWE = 01b) should not be attempted. Behavior after such an operation is undefined and can result in component and/or system hangs.

After sending a one byte write to the Flash, software checks if it can send the next byte to write (check if the write process in the Flash had finished) by reading the Flash Access register. If the bit (FLA.FL_BUSY) in this register is set, the current write did not finish. If bit (FLA.FL_BUSY) is cleared, then software can continue and write the next byte to the Flash.

3.1.3.2.3 Flash Erase Control

When software needs to erase the Flash, it sets bit FLA.FL_ER in the Flash Access register to 1b (Flash Erase) and then set bit EEC.FWE in the EEPROM/Flash Control register to 0b.

Hardware gets this command and sends the erase command to the Flash. Note that the erase process completes automatically. Software should wait for the end of the erase process before any further access to the Flash. This can be checked by using the Flash Write control mechanism.

The op-code used for erase operation is defined in the FLASHOP register.

Sector erase by software is not supported. In order to delete a sector, the serial (bit bang) interface should be used.

3.1.3.2.4 Flash Access Contention

The 82598 implements internal arbitration between Flash accesses initiated through the LAN 0 device and those initiated through the LAN 1 device. If accesses from both LAN devices are initiated during the same approximate size window, the first one is served first and only then the next one. Note that the 82598 does not synchronize between the two entities accessing the Flash though contentions caused from one entity reading and the other modifying the same locations is possible.

To avoid this contention, accesses from both LAN devices should be synchronized using external software synchronization of the memory or I/O transactions responsible for the access. It might be possible to ensure contention-avoidance simply by nature of software sequence.

3.1.4 Network Interface

3.1.4.1 10 GbE Interface

The 82598 provides a complete function supporting 10 Gb/s implementations. The device performs all of the functions required for transmission and reception handling called out in the different standards.

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

The 10 GbE Attachment Unit Interface (XAUI) supports 12.5 Gb/s operations through its four lane differential pairs SerDes transceiver paths. When in XAUI mode, the 82598 provides the full PCS and PMA implementations (through XGXS) including 8b/10b coding, transmit idle randomizer, SerDes, receive synchronization and lanes Deskew.

This interface has 3.125 Gb/s 4-bit data lanes for both receive and transmit. The clock at transmit SerDes operates at 3.125 GHz. The receive circuitry performs the clock and data recovery. After each lane is synchronized, a Deskew mechanism is applied and each lane is aligned properly.

3.1.4.1.1 XGXS – PCS/PMA

The XGMII Extender Sub layer (XGXS) is inserted between the XGMII and XAUI. The source XGXS converts bytes on an XGMII lane into a self clocked, serial, 8b/10b encoded data stream. Each of the four encoded lanes is transmitted across one of the four XAUI lanes (byte striping). The destination XGXS converts the XAUI data stream back into XGMII signals and deskew the four independently clocked XAUI lanes into the single-clock XGMII. The source XGXS converts XGMII Idle control characters into an 8b/10b code_sets. The destination XGXS can add to or delete from the interframe as needed for clock rate disparity compensation prior to converting the interframe code sequence back into XGMII Idle control characters.

XGXS is the common logic components of PCS and PMA in the 10GBASE-X definition. If external serial PMA PHY is attached then XGXS is served as an extender (not the final PCS or PMA functions) from the 82598 to external XGXS component.

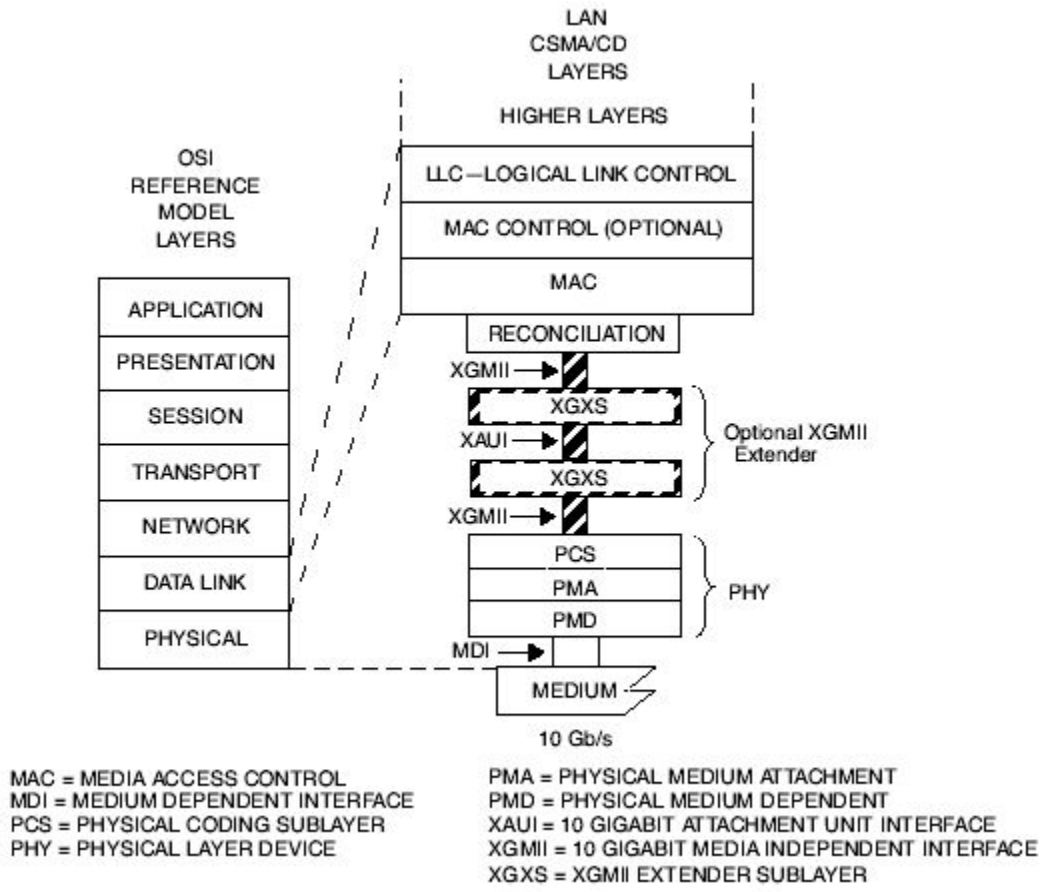


Figure 3-8. LAN CSMA/CD Layers

3.1.4.2 GbE Interface

In addition to supporting 10 Gb/s operations, the 82598 also supports a 1 GbE Interface. To support 1 GbE operation, one of the XAUI Lanes operates at 1.25 Gb/s. All the other 3 XAUI lanes are powered down to electrical idle (output level <50 mV).

3.1.4.3 Auto Negotiation and Link Setup Features

The method for configuring the link between two link partners is highly dependent on the mode of operation as well as the functionality provided by the specific physical layer device.

3.1.4.3.1 Link Configuration

The 82598 network interface meets industry specifications for:

- 10 GbE – XAUI (802.3ae)

- 10 GbE – CX4 (802.3ak)
- 1 GbE backplane
- Ethernet KX (802.3ap)
- Ethernet BX (PICMG3.1)
- 10 GbE backplane
- Ethernet KX4 (802.3ap)

The analog interface is configured to the appropriate electrical specification mode (according to the configuration specified in the EEPROM/AUTOC). Additional analog configuration to the core block is also possible.

3.1.4.3.2 MAC Link Setup and Auto Negotiation

The MAC block in the 82598 supports both 10 Gb/s and 1 Gb/s link modes and the appropriate functionality specified in the standards for these link modes.

Each of these link modes can use different PMD sub-layer and base band medium types.

In 10 Gb/s, there's also support for 10 Gb/s Attachment Unit Interface (XAUI).

Which of these link speeds is used can be determined through static configuration (Force) or Auto Negotiation, as defined in 802.3ap specification clause 73, the auto negotiation process defined in the specification enables the choosing between KX4 (10G) and KX (1G) types.

The 82598 also supports the 1 Gb/s auto negotiation as defined in 802.3 specification clause 37 to support the auto negotiation function when configured to work as Ethernet BX (PICMG).

Link setting is done by configuring the speed configuration and auto negotiation in AUTOC.LMS and restarting auto negotiation by setting AUTOC.RestartAN to 1b.

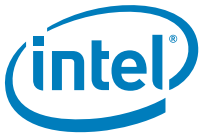
3.1.4.3.3 Hardware Detection of Non-Auto Negotiation Partner

The 82598 also supports parallel detection. Parallel detection is available in parallel to auto negotiation to determine the link mode (KX4 or KX) by activating KX4 and KX alternately and trying to achieve a sync indication from the related PCS this is done as part of the Auto Negotiation to enable link with legacy devices (that do not support Auto Negotiation).

3.1.4.4 MDIO/MDC

3.1.4.4.1 MDIO Direct Access

The Management Data Interface is accessed through registers MSCA and MSRWD. A single management frame is sent by setting bit MSCA.30 to logic 1 and this bit is auto cleared when the frame is done. For old format write operations, the data for the write is first set up in register MSRWD bits 15:0. The next step is to initialize register MSCA with the appropriate control information (start, op code, etc.) and with bit 30 set to logic 1. Bit 30 is reset to logic 0b when both the frame is complete. The steps for old format read operations is identical except that the data in address MSRWD bits (15:0) is ignored and the data read from the external device is stored in register MSRWD bits (31:16). New format operations must be performed in two steps. The address portion of the pair of frames is sent by setting register MSCA bits (15:0) to the desired address, bits (29:28) to 00b which is the start code that identifies new format, and bits (27:26) to 00b which specifies the address portion of the new frame format. A second data frame must be sent after the address frame completes. This second frame is like the old format reads and writes for Op Codes 01b and 10b. Another Op Code of 11b is defined which acts like a read operation except that the external device provides the read data from the register pointed to by the last new format address it used and then the address is incremented.



The output MNG_MDI_INPROG goes high if enabled using register MSCA bit 31 when the command is written to register MSCA bit 30. It stays high until the management frame is complete.

3.1.4.5 Ethernet (Legacy) Flow Control

Flow control as defined in 802.3x, as well as the specific operation of asymmetrical flow control defined by 802.3z, is supported by the 82598. The following four registers are defined for the implementation of flow control:

- Flow Control Receive Thresh High (FCRTH0) – 13-bit high water mark indicating receive buffer fullness
- Flow Control Receive Thresh Low (FCRTL0) – 13-bit low water mark indicating receive buffer emptiness
- Flow Control Transmit Timer Value (FCTTV0) – 16-bit timer value to include in transmitted pause frame
- Flow Control Refresh Threshold Value (FCRTV0) – 16-bit pause refresh threshold value

Flow control is implemented as a means of reducing the possibility of receive buffer overflows, which result in the dropping of received packets, and allows for local controlling of network congestion levels. This might 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.

3.1.4.5.1 MAC Control Frames and Reception of Flow Control Packets

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

1. A match on the six byte multicast address for MAC control frames or to the station address of the device (Receive Address Register 0).
2. A match on the type field.
3. A comparison of the MAC *Control Opcode* field.

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

A value of 0x8808 is compared against the flow control packet's type field to determine if it is a valid flow control packet: XON or XOFF.

The final check for a valid pause frame is the MAC control opcode. At this time only the pause control frame opcode 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 slot time. A slot time is hard wired to 64 byte times, or 512 ns.

XON frame signals the cancellation of the pause from initiated by an XOFF frame. Pause for zero slot times.

The receiver is enabled to receive flow control frames if flow control is enabled via the *RFCE* bit in the FCTRL Register.

Flow control capability must be negotiated between link partners via the auto negotiation process. It is the driver responsibility to reconfigure the flow control configuration after the auto negotiation process was resolved 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 the reception of an XOFF, or PAUSE frame, the device performs the following:

- Increment the appropriate statistics register(s)
- Set the *TXOFF* bit in the TFCS Register
- Initialize the pause timer based on the packet's *Pause Timer* field
- Disable packet transmission or schedule the disabling of transmission after the current packet completes.

Resuming transmission might occur under the following conditions:

- Expiration of the pause timer
- Reception of on XON frame (a frame with its pause timer set to 0b)

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

3.1.4.5.2 Discard Pause Frames and Pass MAC Control Frames

Two bits in the Receive Control register are implemented specifically for control over receipt of pause and MAC control frames. These bits are Discard PAUSE Frames (DPF) and Pass MAC Control Frames (PMCF).

The *DPF* bit forces the discarding of any valid pause frame addressed to the device's station address. If the packet is a valid pause frame and is addressed to the station address (receive address [0]), the device does not pass the packet to host memory if the *DPF* bit is set to logic high. However, if a flow control packet is sent to the station address, and is a valid flow control frame, it is transferred when *DPF* bit is set to 0b. This bit has no affect on pause operation, only the DMA function.

The *PMCF* bit allows for the passing of any valid MAC control frames to the system, which does not have a valid pause opcode. In other words, the frame must have the correct MAC control frame multicast address (or the MAC station address), but does not have the defined pause opcode of 0x0001. Frames of this type are DMA'd to host memory when PMCF is logic high.

3.1.4.5.3 Transmission of Pause Frames

Similar to the reception flow control packets mentioned above, XOFF packets might be transmitted only if this configuration has been negotiated between the link partners via the auto negotiation process. In other words, the setting of this bit by the driver indicates the desired configuration.

The content of the Flow Control Receive Threshold High register determines at what point hardware transmits first a PAUSE frame. Hardware monitors the fullness of the receive FIFO and compares it with the contents of FCRTH. When the threshold is reached, hardware sends a pause frame with its pause time field equal to FCTTV.

At this time, the hardware starts counting an internal shadow counter (reflecting the pause timeout counter at the partner end) from zero. When the counter reaches the value indicated in FCRTV register, then, if the PAUSE condition is still valid (meaning that the buffer fullness is still above the low watermark), an XOFF message is sent again.

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

Hardware sends a pause frame if it has previously sent one and the FIFO overflows. This is intended to minimize the amount of packets dropped if the first pause frame did not reach its target.

3.1.4.6 MAC Speed Change at Different Power Modes

Normal speed negotiation drives to establish a link at the highest possible speed. the The 82598 supports an additional mode of operation, where the MAC drives to establish a link at a low speed. The link-up process allows a link to come up at any possible speed in cases where power is more important than performance. Different behavior is defined for the D0 state and the other non-D0 states.

The 82598 might initiate auto negotiation w/o direct driver command in the following cases:

- When the state of MAIN_PWR_OK pin changes.
- When the MNG_VETO bit value changes.
- On a transition from D0a state to a non-D0a state, or from a non-D0a state to D0a state.

The following figure shows the behavior when going to low power mode.

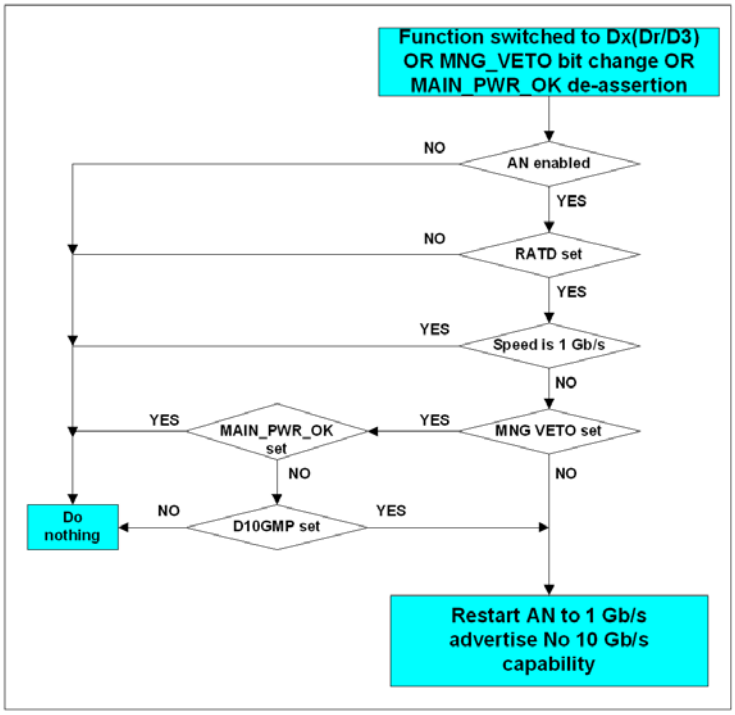


Figure 3-9. Transition to Low-Power Mode

The following figure shows the behavior when going to power-up mode.

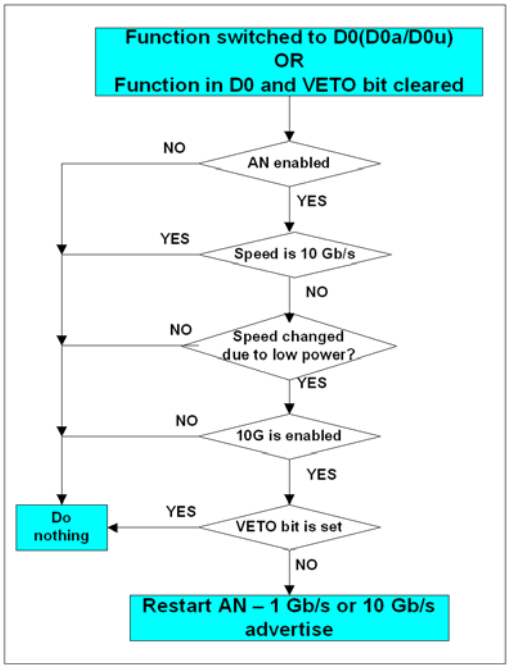


Figure 3-10. Transition to Power-Up Mode

3.2 Initialization

3.2.1 Power Up

3.2.1.1 Power-Up Sequence

The following figure shows the 82598's power-up sequence from power ramp up and until it is ready to accept host commands.

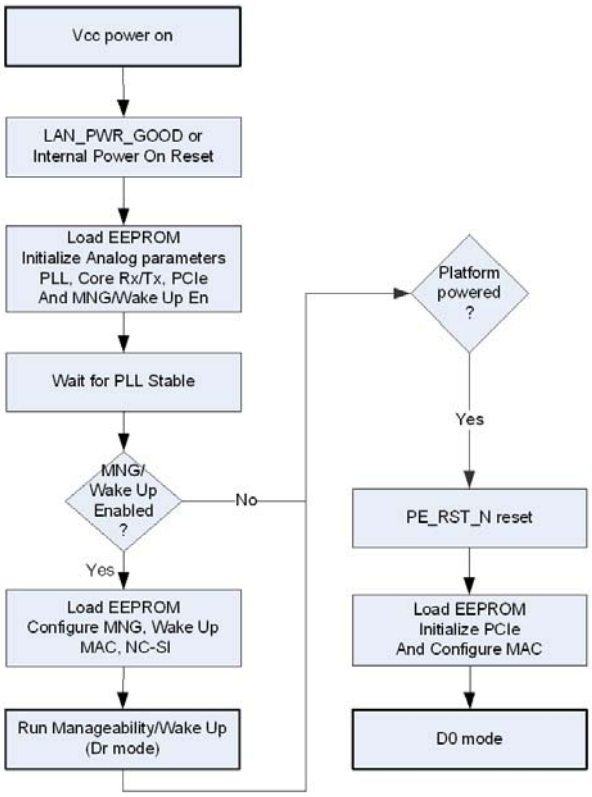


Figure 3-11. 82598 Power-Up Sequence

3.2.1.2 Power-Up Timing Diagram

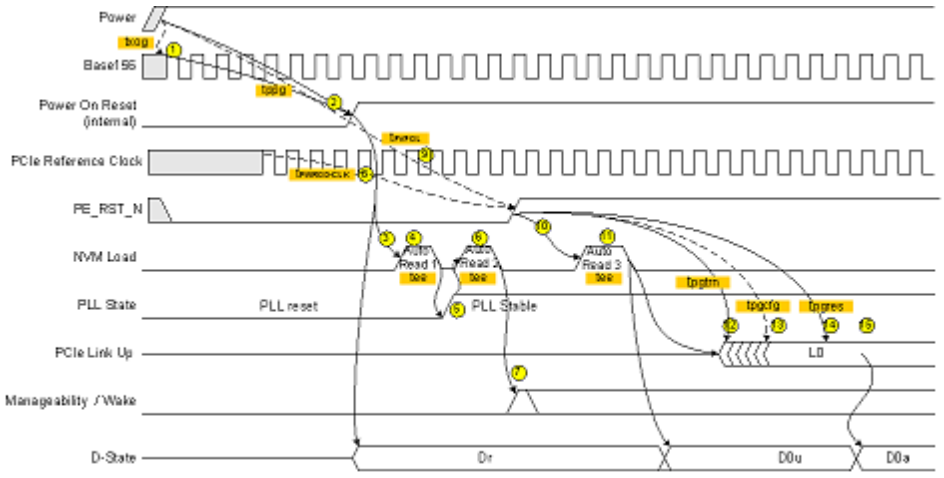


Figure 3-12. Power-Up Timing Diagram

Table 3-35. References for Power-Up Tuning Diagram

Note	Description
1	Base156 clock is stable t_{xog} after the power is stable.
2	Internal reset is released after all power supplies are good and t_{ppg} after Base156 is stable.
3	NVM read starts on the rising edge of internal power on reset or LAN_PWR_GOOD.
4	Sections – EEPROM init and analog configurations are loaded from NVM to configure PLL and core Rx/Tx parameters and to get indication if manageability/wake up are enabled.
5	PLL clock is stable.
6	Sections EEPROM core and EEPROM MAC are read from NVM to configure MAC, manageability and wake up (if manageability /wake up enabled).
7	APM wake up and/or manageability active based on NVM contents (if manageability /wake up enabled).
8	The PCIe reference clock is valid $t_{PWRGD-CLK}$ before de-asserting PE_RST_N (according to PCIe spec).
9	PE_RST_N is de-asserted t_{PVPGL} after power is stable (according to PCIe spec).
10	De-asserting PE_RST_N causes the NVM to be re-read.
11	Sections EEPROM core, EEPROM MAC, PCIe analog, EEPROM PCIe general configuration, and EEPROM PCIe configuration space are read from NVM to configure PCIe and MAC.
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.

Note	Description
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 82598 from D0u to D0 state.

3.2.1.2.1 Timing Requirements

The 82598 requires the following start-up and power state transitions.

Table 3-36. Start-Up and Power-State Transitions

Parameter	Description	Min	Max.	Notes
t_{xog}	Base 156 clock stable from power stable		10 ms	
$t_{PWRGD-CLK}$	PCIe clock valid to PCIe power good	100 µs	-	According to PCIe spec
t_{PVPGL}	Power rails stable to PCIe PWRGD active	100 ms	-	According to PCIe spec
T_{pgcfg}	External PWRGD signal to first configuration cycle.	100 ms		According to PCIe spec

Note: It is assumed that the external 156.25 clock source is stable after the power is applied; the timing for that is part of t_{xog}.

3.2.1.2.2 Timing Guarantees

The 82598 guarantees the following start-up and power state transition related timing parameters.

Table 3-37. Timing Parameters

Parameter	Description	Min	Max.	Notes
t_{xog}	Xosc stable from power stable		10 ms	
t_{ppg}	Internal power good delay from valid power rail	35 ms	35 ms	
t_{ee}	EEPROM read duration		20 ms	
tpgtrn	PCIe PWRGD to start of link training		20 ms	According to PCIe spec
tpgres	External PWRGD to response to first configuration cycle		1 s	According to PCIe spec

3.2.1.3 Reset Operation

The 82598 reset sources are as follows:

- LAN_PWR_GOOD or Internal Power On Reset – The 82598 has an internal mechanism for sensing the power pins. Once the power is up, a stable 82598 creates an internal reset, this reset acts as a master reset of the entire 82598. It is level sensitive, and while it is 0b, holds all registers in reset. LAN_PWR_GOOD or internal power on reset is interpreted as an indication that the 82598 power supplies are all stable. Note that LAN_PWR_GOOD or internal power on reset changes state during system power-up.
- PE_RST_N – Asserting PE_RST_N indicates that both the power and the PCIe clock sources are stable. This pin also asserts an internal reset after a D3cold exit. Most units are reset on the rising edge of PE_RST_N. The only exception is the GIO unit, which is kept in reset while PE_RST_N is de-asserted (level).
- In-band PCIe reset – The 82598 generates an internal reset in response to a PHY 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 controllers.
- D3hot to D0 transition – This is also known as ACPI Reset. The 82598 generates an internal reset on the transition from D3hot power state to D0 (caused after configuration writes from D3 to D0 power state). Note that this reset is per function and resets only the function that transitioned from D3hot to D0.
- Software Reset – Software can reset the 82598 by writing the *Device Reset* bit of the Device Control Register (CTRL.RST). The 82598 re-reads the per-function EEPROM fields after software reset. Bits that are normally read from the EEPROM are reset to their default hardware values. Note that this reset is per function and resets only the function that received the software reset. PCI Configuration space (configuration and mapping) of the 82598 is unaffected. Prior to issuing a software reset, the software device driver needs to execute the master disable algorithm.
- Link Reset – Software can reset the 82598 MAC by writing the *Link Reset* bit of the Device Control Register (CTRL.LRST). The 82598 re-reads the per-function EEPROM fields after link reset. Bits that are normally read from the EEPROM are reset to their default hardware values. Note that this reset is per function and resets only the function that received the link reset. Note that the 82598 executes a software reset each time link reset is asserted. Link reset can also be referred as MAC reset. Prior to issuing link reset, the software device driver needs to execute the master disable algorithm.

- Firmware (FW) Reset – This reset is activated by writing a 1b to the *FWR* bit in the Host Interface Control (HICR) register, or is being asserted by the firmware code or by the internal watchdog expiration.

The resets affect the following registers and logic:

Table 3-38. 82598 Reset Effects

Reset Name	Common Resets			Per Function Resets				
Reset Activation	Internal Power On Reset or LAN_PWR_GOOD	PE_RST_N	In-band PCIe Reset	D3hot? D0	SW Reset	Link Reset	FW Reset	Notes
EEPROM read (global)	X							
EEPROM read (PCIe)		X						
EEPROM read (per function)			X	X	X	X		
LTSSM (back to detect/polling)	X	X	X					
PCIe link data path	X	X	X					
PCI configuration registers RO	X	X	X					
PCI configuration registers R/W	X	X	X	X				
Data path, memory space	X	X	X	X	X	X		2
MAC, PCS, auto negotiation	X	X[6]	X[6]	X[6]	X[6]	X		
Wake up (PM) context	X	X						1,3
Wake up/manageability control/status registers	X							4,5
Manageability unit	X						X	
Strapping pins	X	X	X					

1. If AUX_PWR = 0b the Wakeup Context is reset (PME_Status and PME_En bits should be 0b at reset if the 82598 does not support PME from D3cold).
2. The following register fields do not follow the general rules previously stated:
   a. SDP registers – reset on internal power on reset or LAN_PWR_GOOD only.
   b. LED configuration registers
   c. The *Aux Power Detected* bit in the PCIe Device Status register is reset on internal power on reset, LAN_PWR_GOOD, and PE_RST_N only
   d. FLA – reset on internal power on reset or LAN_PWR_GOOD only.



e. RAH/RAL[n, where n>0], MTA[n], VFTA[n], WUPM[n], FFMT[n], FFVT[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.

3. The wake-up context is defined in the PCI Bus Power Management Interface Specification (sticky bits). It includes:
   a. *PME_En* bit of the Power Management Control/Status Register (PMCSR).
   b. *PME_Status* bit of the Power Management Control/Status Register (PMCSR).
   c. Aux_En in the PCIe registers
   d. The device Requester ID (since it is required for the PM_PME TLP).

Note: The shadow copies of these bits in the Wake Up Control (WUC) register are treated identically.

4. Refers to bits in the WUC register that are not part of the wake-up context (the *PME_En* and *PME_Status* bits).
5. The Wake Up Status (WUS) registers include the following:
   a. WUS register
   b. Wake Up Packet Length (WUPL).
   c. Wake Up Packet Memory (WUPM).
6. The MAC cluster is reset by the appropriate event only if the manageability unit is disabled and the host is in a low power state with WoL disabled.

3.2.2 Specific Function Enable/Disable

3.2.2.1 General

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 a programmer more control over system resource-management, avoid conflicts with add-in NIC solutions, etc. The 82598 provides support for selectively enabling or disabling one or both LAN device(s) in the system.

3.2.2.2 Overview

Device presence (or non-presence) must be established early during BIOS execution, in order to ensure that BIOS resource-allocation (of interrupts, 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 82598 LAN-disable mechanism is implemented in order to be compatible with such a solution. The 82598 samples two pins (pin strapping) on reset to determine the LAN-enable configuration. In addition, the 82598 supports the disabling of one of the PCI functions using EEPROM configuration.

LAN disabling can be done at two different levels. Either the LAN is disabled completely using the LANx_DIS_N pin, or the function is not apparent on the PCIe configuration space using a configuration EEPROM bit. In this case, the LAN function is still available for manageability accesses.

When a particular LAN is fully disabled, all internal clocks to that LAN are disabled. As a result, the 82598 is held in reset and the function presents itself as a dummy device (see Table 3-39).

The sensing of the LANX_Dis_N pins is done after PCIe reset (either PE_RST_N or in-band reset).

As mentioned, one PCI function can be enabled or disabled according to the EEPROM configuration. Two bits in the EEPROM map indicate which function is disabled. An additional EEPROM bit enables the swap between the two LAN functions.

Note: Only one function can be disabled through the EEPROM for manageability use.

It is recommended to keep all the functions at their respective location, even when other functions are disabled. If function #0 (either LAN0 or LAN1) is disabled, then it does not disappear from the PCIe configuration space. Rather, the function presents itself as a dummy function. The device ID and class code of this function changes to other values (dummy function Device ID 0x106A, Class Code 0xFF0000). In addition, the function does not require any memory or I/O space, and does not require an interrupt line.

Table 3-39. PCI Functions Index

PCI Function #	LAN Function Select	Function 0	Function 1	Dummy Function Enable
Both LAN functions are enabled	0	LAN 0	LAN 1	1
Both LAN functions are enabled	1	LAN 1	LAN 0	1
LAN 0 is disabled	0	Dummy	LAN1	1
LAN 0 is disabled (pin)	1	LAN 1	Disable	1
LAN 1 is disabled (pin)	0	LAN 0	Disable	1
LAN 1 is disabled	1	Dummy	LAN 0	1
Both LAN functions are disabled	All PCI functions are disabled entire 82598 is at deep power down			1
Both LAN functions are enabled	0	LAN 0	LAN 1	0
Both LAN functions are enabled	1	LAN 1	LAN 0	0
LAN 0 is disabled	0	LAN 1	Disable	0
LAN 0 is disabled (pin)	1	LAN 1	Disable	0
LAN 1 is disabled (pin)	0	LAN 0	Disable	0
LAN 1 is disabled	1	LAN 0	Disable	0
Both LAN functions are disabled	All PCI functions are disabled entire 82598 is at deep power down			0

3.2.2.3 Event Flow for Enable/Disable Functions

This section describes the driving levels and event sequence for 82598 functionality. Following an internal power on reset/LAN_PWR_GOOD/PE_RST_N/in-band reset, the LANx_DIS_N signals should be driven high (or left open) for nominal operation. If any of the LAN functions are not required statically, its associated disable strapping pin can be tied statically to low.

3.2.2.3.1 BIOS Disable the LAN Function at Boot Time by Using Strapping Option

1. Assume that following a power up sequence LANx_DIS_N signals are driven high.
2. PCIe is established following the PE_RST_N.
3. BIOS recognizes that a LAN function in the 82598 should be disabled.
4. The BIOS drives the LANx_DIS_N signal to the low level.
5. BIOS issues PE_RST_N or an In-Band PCIe reset.

6. As a result, the 82598 samples the LANx_DIS_N signals, disables the LAN function, and issues an internal reset to this function.
7. BIOS might start with the device enumeration procedure (the disabled LAN function is invisible – changed to dummy function).
8. Proceed with normal operation.
9. Re-enabling could be done by driving the LANx_DIS_N signal high and then request the programmer to issue a warm boot to initialize new bus enumeration.

3.2.2.3.2 Multi-Function Advertisement

If one of the LAN devices is disabled and function 0 is the only active function, the 82598 no longer is a multi-function device. The 82598 normally reports a 0x80 in the PCI Configuration Header field *Header Type*, indicating multi-function capability. However, if a LAN ID is disabled and only function 0 is active, the 82598 reports a 0x0 in this field to signify single-function capability.

3.2.2.3.3 Interrupt Use

When both LAN devices are enabled, the 82598 uses both the INTA# and INTB# pins for interrupt reporting. The EEPROM configuration controls which of these two pins are used for each LAN device. The specific interrupt pin used is reported in the PCI Configuration Header *Interrupt Pi*n field associated with each LAN device.

However, if either LAN device is disabled, then the INTA# must be used for the remaining LAN device, therefore the EEPROM configuration must be set accordingly. Under these circumstances, the *Interrupt Pin* field of the PCI Header always reports a value of 0x1, indicating INTA# usage.

3.2.2.3.4 Power Reporting

When both LAN devices are enabled, the PCI Power Management Register Block has the capability of reporting a common power value. The common power value is reflected in the *Data* field of the PCI Power Management registers. The value reported as common power is specified via an EEPROM field and is reflected in the *Data* field each time the *Data_Select* field has a value of 0x8 (0x8 = Common Power Value Select).

When one of LAN ports is disabled and the 82598 appears as a single-function device, the common power value, if selected, reports 0x0 (undefined value), as common power is undefined for a single-function device.

3.2.2.4 Device Disable Overview

When both LAN ports are disabled following an Internal Power on Reset/PE_RST_N/ In-Band reset, the LANx_DIS_N signals should be tied statically to low. At this state the 82598 is disabled, LAN ports are powered down, all internal clocks are shut down, and the PCIe connection is powered down (similar to L2 state).

3.2.2.4.1 BIOS Disable the Device at Boot Time by Using Strapping Option

1. Assume that following a power up sequence LANx_DIS_N signals are driven high.
2. The PCIe is established following the PE_RST_N.
3. BIOS recognizes that the 81598 should be disabled.
4. The BIOS drives the LANx_DIS_N signals to the low level.
5. BIOS issues PE_RST_N or an In-Band PCIe reset.
6. As a result, the 82598 samples the LANx_DIS_N signals, disables the LAN ports, and the PCIe connection.

7. Re-enabling could be done by driving at least one of the LANx_DIS_N signals high and then issue a PE_RST_N to restart the 82598.

3.2.3 Software Initialization and Diagnostics

This section discusses general software notes for the 82598, 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.

3.2.3.1 Power Up State

When the 82598 powers up, it automatically reads the EEPROM. The EEPROM contains sufficient information to bring the link up and configure the 82598 for manageability and/or APM wake up. However, software initialization is required for normal operation.

3.2.3.2 Initialization Sequence

The following sequence of commands is typically issued to the 82598 by the software device driver in order to initialize the 82598 for normal operation. The major initialization steps are:

1. Disable interrupts.
2. Issue a global reset and perform general configuration.
3. Wait for the EEPROM auto read to complete.
4. Wait for a manageability configuration done indication (EEMNGCTL.CFG_DONE).
5. Wait for a DMA init done (RDRXCTL.DMAIDONE)
6. Setup the PHY and the link.
7. Initialize all statistical counters.
8. Initialize receive.
9. Initialize transmit.
10. Enable interrupts.

3.2.3.2.1 Disabling Interrupts During Initialization

Most drivers disable interrupts during initialization to prevent re-entrancy. Interrupts are disabled by writing to the EIMC register. Note that the interrupts need to also be disabled after issuing a global reset, so a typical driver initialization flow is:

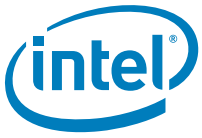
- Disable interrupts
- Issue a global reset
- Disable interrupts (again)

After the initialization completes, a typical driver enables the desired interrupts by writing to the IMS register.

3.2.3.2.2 Global Reset and General Configuration

Device initialization typically starts with a software reset and link reset that puts the 82598 into a known state and enables the device driver to continue the initialization sequence.

Several values in the Device Control (CTRL) register (0x00000/0x00004, RW) need to be set upon power up or after an 82598 reset to normal operation.



To enable flow control, program the MAC Address to RAL and RAH. Other registers to be configured are FCTTV, FCRTL, FCRTH and FCRTV. If flow control is not enabled, the above registers should be written with 0b.

The core configuration according to the electrical specification of the relevant electrical interface should be set prior to the Link setup. This configuration is done through the EEPROM by applying the appropriate settings to the core block.

3.2.3.2.3 Link Setup Mechanisms and Control/Status Bit Summary

3.2.3.2.3.1 BX 1 Gb/s Link Setup

The 82598 PCS initialization is done using the following steps:

1. BX link electrical setup is done according to EEPROM configuration to set the analog interface to the appropriate setting.
2. Configure the *1G Auto Negotiation Enable* in the AUTOC register to make sure the link follows IEEE802.3 clause 37 Auto Negotiation flow.
3. Configure the *Speed Configuration* field to 1 Gb link in the AUTOC register.
4. If necessary, configure any interface fields in the SERDESC register.
5. Configure the *KX/KX4 Auto Negotiation Enable* field to disabled in the AUTOC register. This causes the *Speed Control* field to control the link.
6. Restart the link using the *Restart Auto Negotiation* field in the AUTOC register.
7. Check the link status (sync, link_up, speed) using the LINKS register.

3.2.3.2.3.2 10 Gb/s Link Setup

XAUI / CX4 Link Setup

82598 XAUI/ CX initialization is done using the following steps:

1. XAUI / CX4 link electrical setup is done according to EEPROM configuration to set the analog interface to the appropriate setting.
2. Configure the *Speed Configuration* field to 10 Gb/s link in the AUTOC register.
3. If necessary, configure any interface fields in the SERDESC register.
4. Configure the *KX/KX4 Auto Negotiation Enable* field to disabled in the AUTOC register. This causes the *Speed Control* field to control the link.
5. Restart the link using the *Restart Auto Negotiatio*n field in the AUTOC register.
6. Check the link status (align, link_up, speed) using the Links register.

KX / KX4 Link Setup

82598 KX/KX4 without auto negotiation initialization is done using the following steps:

1. KX/KX4 link electrical setup is done according to EEPROM configuration to set the analog interface to the appropriate setting.
2. Configure the *Speed Configuration* field to 10 Gb/s or 1 Gb/s link in the AUTOC register (for KX4/KX accordingly).
3. If necessary, configure any interface fields in the SERDESC register.
4. Configure the KX/KX4 *Auto Negotiation Enable* field to disabled in the AUTOC register. This causes the *Speed Control* field to control the link.
5. Restart the link using the *Restart Auto Negotiation* field in the AUTOC register.
6. Check the link status (sync, align, link_up, speed) using the LINKS register.

The 82598 KX/KX4 with auto negotiation initialization is done using the following steps:

1. KX / KX4 link electrical setup is done according to EEPROM configuration to set the analog interface to the appropriate setting.
2. If necessary, configure any interface fields in the SERDESC register.
3. Configure the KX/KX4 *Auto Negotiation Enable* field to enabled in the AUTOC register.
4. Configure the *KX_Support* field and any other auto negotiation related fields in the AUTOC register.
5. Restart the link using the *Restart Auto Negotiation* field in the AUTOC register.
6. Check the link status (sync, align, link_up, speed) using the LINKS register.

3.2.3.2.4 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* field in the PCIe Command register).

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

3.2.3.2.5 Receive Initialization

Program the Receive Address Low – RAL (0x05400 + 8*n[n=0..15]; RW) and Receive Address High – RAH (0x05404 + 8*n[n=0..15]; RW) registers with adapter addresses. If an EEPROM is present, RAL0 and RAH0 are loaded from it.

Set up the Multicast Table Array – MTA (0x05200-0x053FC; RW) if reception of Multicast packets is required. The entire table should be zeroed and only desired multicast addresses should be permitted (by writing 0x1 to corresponding bit location). The *MFE* bit should be set in order for multicast filtering to take effect.

Set up the VLAN Filter Table Array – VFTA (0x0A000-0x0A9FC; RW) if VLAN support is required. The entire table should be zeroed and only desired VLAN addresses should be permitted (by writing 0x1 to corresponding bit location). The *VFE* bit should be set in order for VLAN Filtering to take effect.

Working with legacy interrupts:

Program the interrupt mask register to pass any interrupt the software device driver cares about. There is no reason to enable the transmit interrupts.

If software uses the Receive Descriptor Minimum Threshold Interrupt, the *RDMTS* field of RDRXCTL register should be set.

Program the Interrupt Vector allocation table.

Working with MSI-X:

Program the MSI-X table.

The following should be done once per receive queue:

- Allocate a region of memory for the receive descriptor list.
- Receive buffers of appropriate size should be allocated and pointers to these buffers should be stored in the descriptor ring.
- Program the descriptor base address with the address of the region.
- Set the length register to the size of the descriptor ring.

- Program SRRCTL associated with this queue according to the size of the buffers and the required header control.
- If Header Split or Header Replication is required for this queue, the appropriate PSRTYPE must be programmed for the appropriate headers as follows:
  - Program SRRCTL with appropriate values including the *Queue Enable* bit.
  - Set 0b into the tail pointer.
  - Poll the *Queue Enable* bit and make sure that the queue is enabled (read RxDCTL.Qx.25 and make sure it is set).
  - Disable the queue by writing to RxDCTL.Qx.25 = 0b (make sure the descriptor count in the DBU is cleared).
  - Poll the *Queue Enable* bit and make sure that the queue is disabled (read RxDCTL.Qx.25 and make sure it is cleared).
  - Enable the queue by writing to RxDCTL.Qx.25 = 1b.
  - Poll the *Queue Enable* bit and make sure that the queue is enabled (read RxDCTL.Qx.25 and make sure it is set).
- Program RXDCTL with appropriate values including the *Queue Enable* bit.
- Program the tail pointer to enable the fetch of descriptors

Note: Packets to a disabled queue are dropped.

3.2.3.2.6 Dynamic Enabling and Disabling of Receive Queues

Receive queues can be enabled or disabled dynamically if the following procedure is followed.

1. Enabling:
   a. Follow the per queue initialization described in the previous section.
2. Disabling:
   a. Disable the direction of packets to this queue.
   b. Disable the queue by clearing the enable bit in RXDCTL. The 82598 stops fetching and writing back descriptors from this queue. Any further packet that is directed to this queue is dropped. If a packet is being processed, the 82598 completes the current buffer write if the packet spreads over more than one buffer. All subsequent buffers are not written.
   c. The 82598 clears the *RXDCTL.ENABLE* bit only after all pending memory accesses to the descriptor ring are done. The software device driver should poll this bit and then wait an additional amount of time (> 100 μs) before releasing the memory allocated to this queue.

There might be additional packets in the receive packet buffer targeted to the disabled queue. The arbitration might be such that it would take a long time to drain down those packets. If software re-enables a queue before all packets to that queue were drained, the enabled queue might potentially get packets directed to the old configuration of the queue. For example, VM goes down and a different VM gets the queue (if there were undrained packets) these packets targeted to the previous VM would get to the new VM that owns the queue.

The receive path can be disabled only after all the receive queues are disabled.

3.2.3.2.7 Transmit Initialization

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.
- Initialize the transmit descriptor registers (TDBAL, TDBAH, TDL).
- Program the Transmit Descriptor Control registers with the desired TX descriptor write back policy. Suggested values are:
  - WTHRESH = 1b
  - All other fields 0b.
  - Enable queue using TXDCTL.ENABLE
  - Poll the *Queue Enable* bit to make sure the queue is enabled (read TXDCTL.Qx.25; check that it is set).

3.2.3.2.8 Dynamic Enabling and Disabling of Transmit Queues

Transmit queues can be enabled or disabled dynamically if the following procedure is followed.

1. Enabling:
   a. Follow the per queue initialization described in the previous section.
2. Disabling:
   a. Stop storing packets for transmission in this queue.
   b. Wait until the head of the queue TDH equals the tail TDT – indicates the queue is empty.
   c. Wait until all descriptors are written back (polling DD bit in ring or polling the Head_WB content). It might be required to flush the transmit queue by setting the TXDCTL[n].SWFLSH if the RS bit in the last fetched descriptor is not set or if WTHRESH is greater than zero.
   d. Disable the queue by clearing TXDCTL.ENABLE.

The transmit path can be disabled only after all transmit queue are disabled.

3.3 Power Management and Delivery

This section describes how power management is implemented in the 82598.

3.3.1 Power Delivery

The 82598's power is delivered through external voltage regulators. Refer to Section 8. for more details.

3.3.1.1 82598 Power States

The 82598 supports the D0 and D3 power states defined in the PCI power management and PCIe specifications. D0 is divided into two sub-states: D0u (D0 Un-initialized) and D0a (D0 active). In addition, the 82598 supports a Dr state that is entered when PE_RST_N is asserted (including the D3cold state).

Figure 3-13 shows the power states and transitions between them.

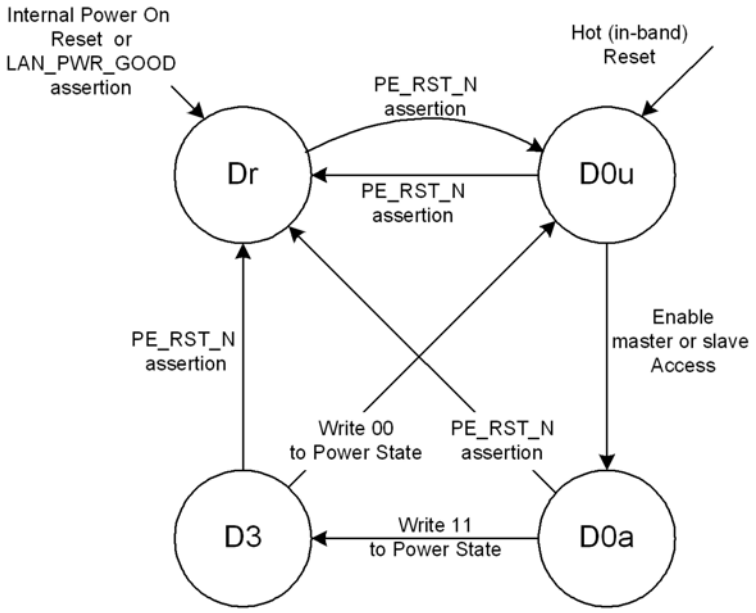


Figure 3-13. Power Management State Diagram

3.3.1.2 Auxiliary Power Usage

If ADVD3WUC=1b, the 82598 uses the AUX_PWR indication that auxiliary power is available to the 82598, and therefore advertises D3cold Wake Up support. The amount of power required for the function (which includes the entire NIC) is advertised in the Power Management Data register, which is loaded from the EEPROM.

If D3cold is supported, the *PME_En* and *PME_Status* bits of the Power Management Control/Status register (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).

The only effect of setting AUX_PWR to 1b is advertising D3cold Wake Up support and changing the reset function of PME_En and PME_Status. AUX_PWR is a strapping option in the 82598.

The 82598 tracks the *PME_En* bit of the Power Management Control/Status register (PMCSR) and the *Auxiliary (AUX) Power PM Enable* bit of the PCIe Device Control register to determine the power it might consume (and therefore its power state) in the D3cold state (internal Dr state). Note that the actual amount of power differs between form factors.

The *PCIE_Aux* bit in the EEPROM determines if the 82598 complies with the auxiliary power regime defined in the PCIe specification. If set, the 82598 might consume higher aux power according to the following settings:

- If the *Auxiliary (AUX) Power PM Enable* bit of the PCIe Device Control register is set, the 82598 might consume higher power for any purpose (even if PME_En is not set).
- If the *Auxiliary (AUX) Power PM Enable* bit of the PCIe Device Control register is cleared, higher power consumption is determined by the *PCI-PM legacy PME_En* bit of the Power Management Control/Status register (PMCSR).

If the *PCIE_Aux* bit in the EEPROM is cleared, the 82598 consumed aux power in Dr state independent of the setting of either the *PME_En* bit or the *Auxiliary (AUX) Power PM Enable* bit.

3.3.1.3 Interconnects Power Management

This section describes the power reduction techniques used by the 82598's main interconnects.

3.3.1.3.1 PCIe Link Power Management

The PCIe link state follows the power management state of the 82598. Since the 82598 incorporates multiple PCI functions, the device power management state is defined as the power management state of the most awake function:

- If any function is in D0 state (either D0a or D0u), the PCIe link assumes the 82598 is in D0 state.

Else:

- If the functions are in D3 state, the PCIe link assumes the 82598 is in D3 state.

Else:

- The device is in Dr state (PE_RST_N is asserted to all functions).

The 82598 supports all PCIe power management link states other than L1 ASPM:

- L0 state is used in D0u and D0a states.
- The L0s state is used in D0a and D0u states each time the link conditions apply.
- The L1 state is 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 also if PME is not enabled in other Dr transitions.

The 82598 support for Active State Link Power Management is reported via the PCIe Active State Link PM Support register loaded from EEPROM.

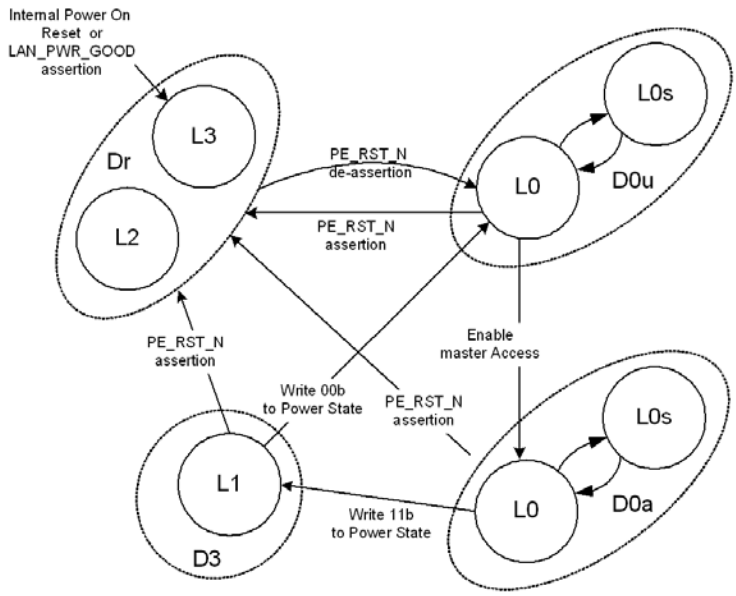


Figure 3-14. Link Power Management State Diagram

While in L0 state, the 82598 transitions the transmit lane(s) into L0s state once the idle conditions are met for a period of time defined below.

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.
- The *L0S_ENTRY_LAT* bit in the PCIe Control (GCR) register, determines l0s entry latency. When set to 0b, L0s entry latency is the same as L0s exit latency of the 82598 at the other end of the link. When set to 1b, L0s entry latency is (L0s exit latency of the 82598 at the other end of the link/4). The default value is 0b (entry latency is the same as L0s exit latency of the 82598 at the other end of the link).
- L0s exit latency (as published in the *L0s Exit Latency* field of the Link Capabilities register) is loaded from EEPROM. Separate values are loaded when the 82598 shares the same reference PCIe clock with its partner across the link and when the 82598 uses a different reference clock than its partner across the link. The 82598 reports whether it uses the slot clock configuration through the *PCIe Slot Clock Configuration* bit loaded from the *Slot_Clock_Cfg* EEPROM bit.
- L0s Acceptable Latency (as published in the *Endpoint L0s Acceptable Latency* field of the Device Capabilities register) is loaded from EEPROM.

3.3.1.3.2 Network Interfaces Power Management

The 82598 transitions any of the XAUI interfaces into a low-power state in the following cases:

- The respective LAN function is in LAN disable mode using the LANx_DIS_N pin.
- The 82598 is in Dr State, APM WoL is disabled for the port, ACPI wake is disabled for the port and pass-through manageability is disabled for the port.

Use of the LAN ports for pass-through manageability follows the following behavior:

- If manageability is disabled (as loaded from the EEPROM), then LAN ports are not allocated for manageability.
- If manageability is enabled:
  - Power-up – Following EEPROM read, a single port is enabled for manageability, running at the lowest speed supported by the interface. If APM WoL is enabled on a single port, the same port is used for manageability. Otherwise, manageability protocols (teaming) determine which port is used.
  - D0 state – Both LAN ports are enabled for manageability.
  - D3 and Dr states – A single port is enabled for manageability, running at the lowest speed supported by the interface. If WoL is enabled on a single port, the same port is used for manageability. Otherwise, manageability protocols (such as teaming) determine which port is used.

Enabling a port as a result of the above causes an internal reset of the port.

When a XAUI interface is in low-power state, the 82598 asserts the respective PHY0_PWRDN_N or PHY1_PWRDN_N pin to enable an external PHY device to power down as well.

3.3.1.4 Power States

3.3.1.4.1 D0 Uninitialized State

The D0u state is a low-power state used after PE_RST_N is de-asserted following power up (cold or warm), on hot reset (in-band reset through PCIe physical layer message) or on D3 exit.

When entering D0u, the 82598 disables Wake ups. If the *APM Mode* bit in the EEPROM's Control Word 3 is set, then APM Wake Up is enabled.

3.3.1.4.1.1 Entry into D0u State

D0u is reached from either the Dr state (on de-assertion of internal 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 the internal PE_RST_N means that the entire state of the 82598 is cleared, other than sticky bits. State is loaded from the EEPROM, followed by establishment of the PCIe link. Once this is done, configuration software can access the 82598.

On a transition from D3 to D0u state, the 82598 requires that software perform a full re-initialization of the function including its PCI configuration space.

3.3.1.4.2 D0active State

Once memory space is enabled, the 82598 enters an active state. It can transmit and receive packets if properly configured by the driver. Any APM Wakeup previously active remains active. The driver can deactivate APM Wakeup by writing to the Wake Up Control (WUC) register, or activate other wake up filters by writing to the Wake Up Filter Control (WUFC) register.

3.3.1.4.2.1 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. The DMA, MAC, and PHY of the appropriate LAN function are enabled.

3.3.1.4.3 D3 State (PCI-PM D3hot)

The 82598 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 reset state are maintained. Upon transitioning to D3 state, the 82598 clears the *Memory Access Enable* and *I/O Access Enable* bits of the PCI Command register, which disables memory access decode. In D3, the 82598 only responds to PCI configuration accesses and does not generate master cycles.

Configuration and message requests are the only PCIe TLPs accepted by a function in the D3hot state. All other received requests must be handled as unsupported requests, and all received completions can optionally be handled as unexpected completions. If an error caused by a received TLP (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 v2.0 (2.5 GT/s) Specification.

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.

3.3.1.4.3.1 Entry to D3 State

Transition to D3 state is through a configuration write to the *Power State* field of the PCI-PM registers.

Prior to transition from D0 to the D3 state, the software device driver disables scheduling of further tasks to the 82598; it masks all interrupts, it does not write to the Transmit Descriptor Tail register or to the Receive Descriptor Tail register and operates the master disable algorithm as defined in Section 3.3.1.4.3.2. If wake-up capability is needed, the software device driver should set up the appropriate wake-up registers and the system should write a 1b to the *PME_En* bit of the Power Management Control/Status register (PMCSR) or to the *Auxiliary (AUX) Power PM Enable* bit of the PCIe Device Control register prior to the transition to D3.

If all PCI functions are programmed into D3 state, the 82598 brings its PCIe link into the L1 link state. As part of the transition into L1 state, the 82598 suspends scheduling of new TLPs and waits for the completion of all previous TLPs it has sent. The 82598 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 is kept in the 82598 (and discarded later on D3 exit). Any transmit packets that have not be sent can still be transmitted (assuming the Ethernet link is up).

In preparation to a possible transition to D3cold state, the software device driver can disable one of the LAN ports (LAN disable) and/or transition the link(s) to Gb speed (if supported by the network interface). See Section 3.3.1.3.2 for a description of network interface behavior in this case.

3.3.1.4.3.2 Master Disable

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 82598 must not issue master accesses for this function. Due to the full-duplex nature of PCIe, and the pipelined design in the 82598, it might happen that multiple requests from several functions 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 device 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 82598 blocks new master requests by this function. The 82598 then proceeds to issue any pending requests by this function. This bit is cleared on master reset (Internal Power On Reset all the way to software reset) to enable master accesses.
- *GIO Master Enable Status* bits in the Device Status register – Cleared by the 82598 when the *GIO Master Disable* bit is set and no master requests are pending by the relevant function. 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 82598 clears the *GIO Master Enable Status* bit:
- Master requests by the transmit and receive engines
- All pending completions to the 82598 are received.

Notes:

- The software device driver sets the *GIO Master Disable* bit when notified of a pending master disable (or D3 entry). The 82598 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 master request to the PCIe link. Can be done either through reset or by the software device driver.

3.3.1.4.4 Dr State

Transition to Dr state is initiated on several occasions:

- On system power up – Dr state begins with the assertion of Internal Power On Reset or LAN_PWR_GOOD 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 82598 into the Dr state by asserting PCIe PE_RST_N.

Any wake-up 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 82598 might maintain functionality (for WoL or manageability) or might enter a Dr Disable state (if no WoL and no manageability) for minimal 82598 power.

3.3.1.4.4.1 Dr Disable Mode

The 82598 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 82598 (all PCI functions) is in Dr state
- APM WOL is inactive for both LAN functions
- Pass-through manageability is disabled

- ACPI PME is disabled for all PCI functions

Entry into Dr Disable is done on assertion of PCIe PE_RST_N. It might also be possible to enter Dr Disable mode by reading the EEPROM while already in Dr state. The usage model for this later case is on system power up, assuming that manageability and wake up are not required. Once the 82598 enters Dr state on power-up, the EEPROM is read. If the EEPROM contents determine that the conditions to enter Dr Disable are met, the 82598 then enters this mode (assuming that PCIe PE_RST_N is still asserted).

Exit from Dr Disable is through de-assertion of PCIe PE_RST_N.

If Dr Disable mode is entered from D3 state, the 82598 asserts the DEV_PWRDN_N output signal to indicate to the platform that it might remove power from the 82598. The platform must remove all power rails from the 82598 if it needs to use this capability. Exiting from this state is through power-up reset to the 82598. Note that the state of the DEV_PWRDN_N and the PHYx_PWRDN_N outputs is undefined once power is removed from the 82598.

3.3.1.4.4.2 Entry to Dr State

Dr entry on platform power-up is as follows:

- Asserting Internal Power On Reset or LAN_PWR_GOOD. The 82598 power is kept to a minimum by keeping the XAUI interfaces in low power.
- The EEPROM is then read and determines the 82598 configuration.
- If the *APM Enable* bit in the EEPROM's Initialization Control Word 2 is set then APM wake up is enabled (for each port independently).
- If the *MNG Enable* bit in the EEPROM is set, pass-through manageability is not enabled.
- Each of the LAN ports can be enabled, if required, for WoL or manageability. See Section 3.3.1.3.2 for exact condition to enable a port.
- The PCIe link is not enabled in Dr state following system power up (since PE_RST_N is asserted).

Entry to Dr state from D0a state is through assertion of the PE_RST_N signal. An ACPI transition to the G2/S5 state is reflected in an 82598 transition from D0a to Dr state. The transition might be orderly (programmer selected a show down operating system option), in which case the software device driver might have a chance to intervene. Or, it might be an emergency transition (power button override), in which case, the software device driver is not notified.

Transition from D3 state to Dr state is done by assertion of PE_RST_N signal. 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.

3.3.1.5 Timing of Power-State Transitions

The following sections give detailed timing for the state transitions. In the diagrams the dotted connecting lines represent the 82598 requirements, while the solid connecting lines represent the 82598 guarantees.

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

3.3.1.5.1 Transition from D0a to D3 and back without PE_RST_N

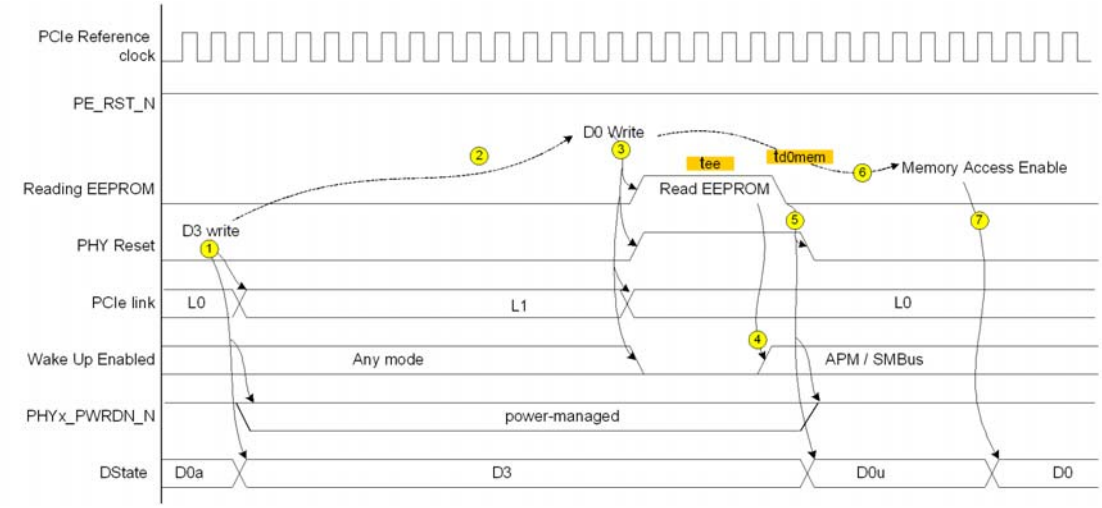


Figure 3-15. D0a to D3 and back without PE_RST_N

Table 3-40. 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 82598 to D3.
2	The system can keep the 82598 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 Power Management Control/Status Register (PMCSR).
4	APM wake up or manageability can be enabled based on what is read in the EEPROM.
5	After reading the EEPROM, the LAN ports are enabled and the 82598 transitions to D0u state.
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 82598 from D0u to D0 state.

3.3.1.5.2 Transition from D0a to D3 and Back with PE_RST_N

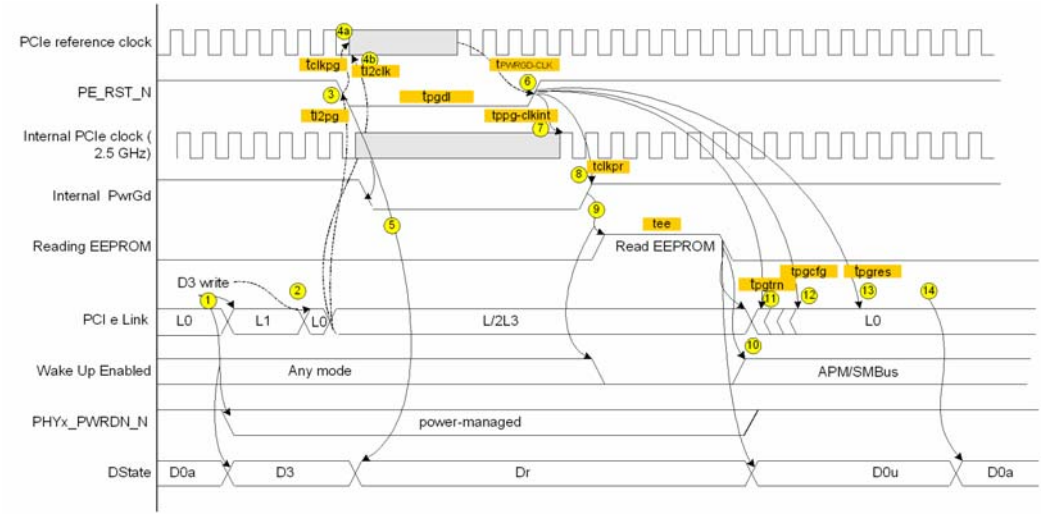


Figure 3-16. D0a to D3 and Back with PE_RST_N

Table 3-41. D0a to D3 and Back with PE_RST_N

Note	
1	Writing 11b to the *Power State* field of the Power Management Control/Status Register (PMCSR) transitions the 82598 to D3. PCIe link transitions to L1 state.
2	The system can delay an arbitrary amount of time between setting D3 mode and transitioning the link to an 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 82598 transitions to Dr state.
6	The system starts the PCIe reference clock $t_{PWRGD-CLK}$ 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 tclkpr after the external PE_RST_N signal
9	Assertion of internal PCIe PWRGD causes the EEPROM to be re-readand disables wake up.
10	APM wake up mode can be enabled based on what is read from the EEPROM.
11	Link training starts after t_{pgtrn} from PE_RST_N de-assertion.
12	A first PCIe configuration access can arrive after t_{pgcfg} from PE_RST_N de-assertion.
13	A first PCI configuration response can be sent after t_{pgres} from PE_RST_N de-assertion
14	Writing a 1b to the *Memory Access Enable* bit in the PCI Command register transitions the 82598 from D0u to D0 state.

3.3.1.5.3 Transition from D0a to Dr and Back Without Transition to D3

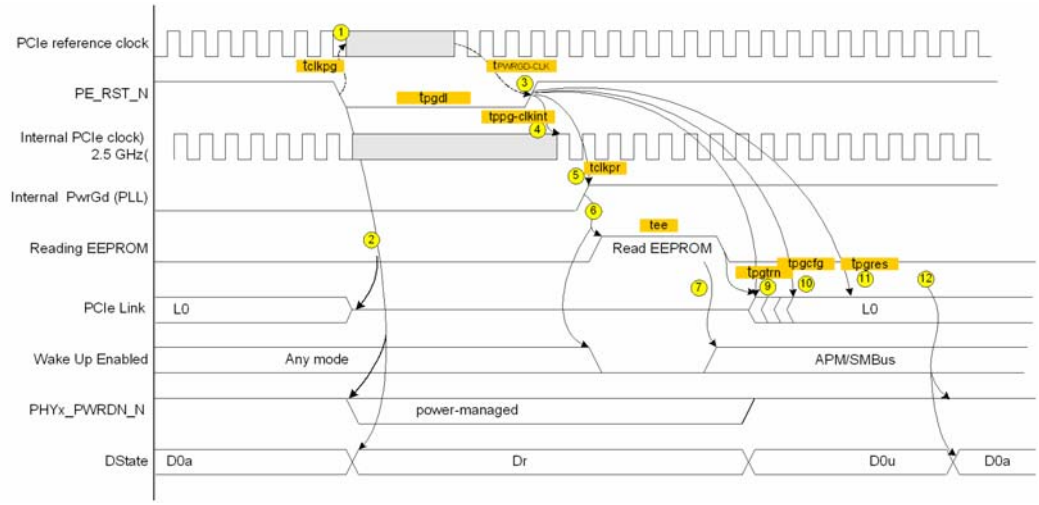


Figure 3-17. D0a to Dr and Back Without Transition to D3

Table 3-42. 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 82598 transitions to Dr state and the PCIe link transition to electrical idle.
3	The system starts the PCIe reference clock $t_{PWRGD-CLK}$ 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	Assertion of internal PCIe PWRGD causes the EEPROM to be re-read and disables wake up.
7	APM wake up mode can be enabled based on what is read from the EEPROM.
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 82598 from D0u to D0 state.

3.3.1.5.4 Timing Requirements

The 82598 requires the following start-up and power-state transitions.

Table 3-43. Start-Up and Power-State Transitions

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

3.3.1.5.5 Timing Guarantees

The 82598 guarantees the following start-up and power-state transition related timing parameters.

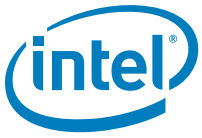


Table 3-44. Start-up and Power-State Transition Related Timing Parameters

Parameter	Description	Min	Max.	Notes
t_{xog}	Xosc stable from power stable		10 ms	
t_{ppg}	Internal power good delay from valid power rail	35 ms	35 ms	
t_{ee}	EEPROM read duration		20 ms	
$t_{ppg\text{-}clkint}$	PCIe PWRGD to internal PLL lock	-	50 µs	
t_{clkpr}	Internal PCIe PWGD from external PCIe PWRGD		50 µs	
t_{pgtrn}	PCIe PWRGD to start of link training		20 ms	According to PCIe specification.
t_{pgres}	External PWRGD to response to first configuration cycle		1 s	According to PCIe specification.

3.3.2 Wake Up

3.3.2.1 Advanced Power Management Wake Up

Advanced Power Management Wake Up, or APM Wake Up, was previously known as Wake on LAN (WoL). It is a feature that has existed in the 10/100 Mb/s NICs for several generations. The basic premise is to receive a broadcast or unicast packet with an explicit data pattern, and then to assert a signal to wake up the system. In the earlier generations, this was accomplished by using a special signal that ran across a cable to a defined connector on the motherboard. The NIC asserts the signal for approximately 50 ms to signal a wake up. The 82598 uses (if configured to) an in-band PM_PME message for this.

At power-up, the 82598 reads the *APM Enable* bit from the EEPROM into the *APM Enable* (APME) bits of the GRC register This bit controls the enabling of APM wake up.

When APM Wakeup is enabled, the 82598 checks all incoming packets for Magic Packets*. Refer to Section 3.3.2.3.1.4 for more information.

Once the 82598 receives a matching magic packet, it:

- Sets the *PME_Status* bit in the Power Management Control/Status Register (PMCSR) and issues a PM_PME message (in some cases, this might be required to assert the WAKE# signal first to resume power and clock to the PCIe interface).
- Stores the first 128 bytes of the packet in WUPM.
- Sets the *Magic Packet Received* bit in the WUS register.
- Sets the packet length in the WUPL register.

The 82598 maintains the first magic packet received in WUPM until the software device driver writes a 0b to the *Magic Packet Received MAG* bit in the WUS register.

APM wake up is supported in all power states and only disabled if a subsequent EEPROM read results in the *APM Wake Up* bit being cleared or the software explicitly writes a 0b to the *APM Wake Up* (APM) bit of the GRC register.

3.3.2.2 ACPI Power Management Wakeup

The 82598 supports ACPI power management based wake ups. It can generate system wake-up events from three sources:

- Reception of a Magic Packet*.
- Reception of a network wake-up packet.
- Detection of a link change of state.

Activating ACPI power management wake up requires the following steps:

- The operating system (at configuration time) writes a 1b to the *Pme_En* bit (bit 8) of the PMCSR register.
- The software device driver clears all pending wake-up status in the WUS register by writing 1b to all the status bits.
- The software device driver programs the Wake Up Filter Control (WUFC) register to indicate the packets it needs to wake up and supplies the necessary data to the IPv4/v6 Address Table (IP4AT, IP6AT), Flexible Host Filter Table (FHFT). It can also set the *Link Status Change Wake Up Enable* (LNKC) bit in the WUFC register to cause a wake up when the link changes state.
- Once the 82598 wakes up the system the driver needs to clear WUS and WUFC until the next time the system goes to a low power state with wake up.

Normally, after enabling wake up, the operating system writes (11b) to the lower two bits of the PMCSR to put the 82598 into low-power mode.

Once wake up is enabled, the 82598 monitors incoming packets, first filtering them according to its standard address filtering method, then filtering them with all of the enabled wakeup filters. If a packet passes both the standard address filtering and at least one of the enabled wake-up filters, the 82598:

- Sets the *PME_Status bit* in the PMCSR.
- If the *PME_En* bit in the PMCSR is set, asserts PE_WAKE_N.
- Stores the first 128 bytes of the packet in WUPM.
- Sets one or more of the *Received* bits in the WUS register (the 82598 sets more than one bit if a packet matches more than one filter).
- Sets the packet length in the WUPL register.

If enabled, a link state change wakeup causes similar results, setting *PME_Status*, asserting PE_WAKE_N and setting the *Link Status Changed* (LNKC) bit in the WUS register when the link goes up or down.

PE_WAKE_N remains asserted until the operating system either writes a 1b to the *PME_Status* bit of the PMCSR register or writes a 0b to the *PME_En* bit.

After receiving a wakeup packet, the 82598 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.

3.3.2.3 Wake-Up Packets

The 82598 supports various wake-up packets using two types of filters:

- Pre-defined filters

- Flexible filters

Each of these filters are enabled if the corresponding bit in the WUFC register is set to 1b.

When VLAN filtering is enabled, a packet that passed any of the receive wake-up filters should only cause a wake-up event if it also passed VLAN filtering. This is true for all wake-up packets except for directed packets (including exact, multicast indexed, and broadcast) and magic packets, which are not broadcaster.

3.3.2.3.1 Pre-Defined Filters

The following packets are supported by the 82598's pre-defined filters:

- Directed packet (including exact, multicast indexed, and broadcast)
- Magic Packet*
- ARP/IPv4 request packet
- Directed IPv4 packet
- Directed IPv6 packet

Each of these filters are enabled if the corresponding bit in the WUFC register is set to 1b.

The explanation of each filter includes a table showing which bytes at which offsets are compared to determine if the packet passes the filter. Both VLAN frames and LLC/Snap can increase the given offsets if they are present.

3.3.2.3.1.1 Directed Exact Packet

The 82598 generates a wake-up event after receiving any packet whose destination address matches one of the 16 valid programmed receive addresses if the *Directed Exact Wake Up Enable* bit is set in the Wake Up Filter Control (WUFC.EX) register.

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	Match any pre-programmed address

3.3.2.3.1.2 Directed Multicast Packet

For multicast packets, the upper bits of the incoming packet's destination address index a bit vector, the Multicast Table Array that indicates whether to accept the packet. If the *Directed Multicast Wake Up Enable* bit set in the Wake Up Filter Control (WUFC.MC) register and the indexed bit in the vector is 1b then the 82598 generates a wake-up event. The exact bits used in the comparison are programmed by software in the *Multicast Offset* field of the Multicast Control (MCSTCTRL.MO) register.

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	See 3.3.2.3.1.2

3.3.2.3.1.3 Broadcast

If the *Broadcast Wake Up Enable* bit in the Wake Up Filter Control (WUFC.BC) register is set, the 82598 generates a wake-up event when it receives a broadcast packet.

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address	0xFF*6	Compare	

3.3.2.3.1.4 Magic Packet*

Magic packets are defined in:

http://www.amd.com/products/npd/overview/20212.html as:

Magic Packet Technology Details

Once the LAN controller has been put into the Magic Packet mode, it scans all incoming frames addressed to the node for a specific data sequence, which indicates to the controller that this is a Magic Packet frame. A Magic Packet frame must also meet the basic requirements for the LAN technology chosen, such as SOURCE ADDRESS, DESTINATION ADDRESS (which may be the receiving station's IEEE address or a MULTICAST address which includes the BROADCAST address), and CRC. The specific data sequence consists of 16 duplications of the IEEE address of this node, with no breaks or interruptions. This sequence can be located anywhere within the packet, but must be preceded by a synchronization stream. The synchronization stream allows the scanning state machine to be much simpler. The synchronization stream is defined as 6 bytes of FFh. The device also accepts a BROADCAST frame, as long as the 16 duplications of the IEEE address match the address of the machine to be awakened.

The 82598 expects the destination address to:

1. Be the broadcast address (0xFF.FF.FF.FF.FF.FF)
2. Match the value in Receive Address register 0 (RAH0, RAL0). This is initially loaded from the EEPROM but might be changed by the software device driver.
3. Match any other address filtering enabled by the software device driver.

The 82598 searches for the contents of Receive Address register 0 (RAH0, RAL0) as the embedded IEEE address (it catches the case of seven 0xFFs followed by the IEEE address). As soon as one of the first 96 bytes after a string of 0xFFs doesn't match, it continues to search for anther set of at least six 0xFFs followed by the 16 copies of the IEEE address later in the packet.

A Magic Packet's destination address must match the address filtering enabled in the configuration registers with the exception that broadcast packets are considered to match even if the *Broadcast Accept* bit of the Receive Control register (FCTRL.BAM) is 0b. If *APM Wakeup* is enabled in the EEPROM, the 82598 starts up with the Receive Address register 0 (RAH0, RAL0) loaded from the EEPROM. This enables the 82598 to accept packets with the matching IEEE address before the software device driver comes up.

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	8	Possible LLC/SNAP Header		Skip	
12	4	Possible VLAN Tag		Skip	

12	4	Type		Skip	
Any	6	Synchronizing Stream	0xFF*6+	Compare	
any+6	96	16 Copies of Node Address	0xA*16	Compare	Compared to Receive Address Register 0 (RAH0, RAL0)

3.3.2.3.1.5 ARP/IPv4 Request Packet

The 82598 supports receiving ARP Request packets for wakeup if the *ARP* bit is set in the Wake Up Filter Control WUFC) register. Four IPv4 addresses are supported and are programmed in the IPv4 Address Table (IP4AT). A successfully matched packet must pass L2 address filtering, a Protocol Type of 0x0806, an ARP OPCODE of 0x01, and one of the four programmed IPv4 addresses. The 82598 also handles ARP Request packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

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	8	Possible LLC/SNAP Header		Skip	
12	4	Possible VLAN Tag		Compare	
12	2	Type	0x0806	Compare	ARP
14	2	Hardware Type	0x0001	Compare	
16	2	Protocol Type	0x0800	Compare	
18	1	Hardware Size	0x06	Compare	
19	1	Protocol Address Length	0x04	Compare	
20	2	Operation	0x0001	Compare	
22	6	Sender Hardware Address	-	Ignore	
28	4	Sender IP Address	-	Ignore	
Offset	**# of Bytes**	**Field**	**Value**	**Action**	**Comment**
32	6	Target Hardware Address	-	Ignore	
38	4	Target IP Address	IP4AT	Compare	Might match any of four values in IP4AT

3.3.2.3.1.6 Directed IPv4 Packet

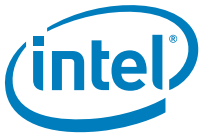
The 82598 supports receiving Directed IPv4 packets for wakeup if the *IPV4* bit is set in the WUFC register. Four IPv4 addresses are supported and are programmed in the IPv4 Address Table (IP4AT). A successfully matched packet must pass L2 address filtering, a Protocol Type of 0x0800, and one of the four programmed IPv4 addresses. The 82598 also handles Directed IPv4 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

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	8	Possible LLC/SNAP Header		Skip	
12	4	Possible VLAN Tag		Compare	
12	2	Type	0x0800	Compare	IP
14	1	Version/ HDR length	0x4X	Compare	Check IPv4
15	1	Type of Service	-	Ignore	
16	2	Packet Length	-	Ignore	
18	2	Identification	-	Ignore	
20	2	Fragment Info	-	Ignore	
22	1	Time to live	-	Ignore	
23	1	Protocol	-	Ignore	
24	2	Header Checksum	-	Ignore	
26	4	Source IP Address	-	Ignore	
30	4	Destination IP Address	IP4AT	Compare	May match any of 4 values in IP4AT

3.3.2.3.1.7 Directed IPv6 Packet

The 82598 supports receiving Directed IPv6 packets for wakeup if the IPv6 bit is set in the Wake Up Filter Control (WUFC) register. One IPv6 address is supported is programmed in the IPv6 Address Table (IP6AT). A successfully matched packet must pass L2 address filtering, a Protocol Type of 0x08DD, and the programmed IPv6 address. The 82598 also handles Directed IPpv6 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

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	8	Possible LLC/SNAP Header		Skip	
12	4	Possible VLAN Tag		Compare	
12	2	Type	0x08DD	Compare	IP
14	1	Version/ Priority	0x6X	Compare	Check IPv6
15	3	Flow Label	-	Ignore	
18	2	Payload Length	-	Ignore	
20	1	Next Header	-	Ignore	
21	1	Hop Limit	-	Ignore	
22	16	Source IP Address	-	Ignore	
38	16	Destination IP Address	IP6AT	Compare	Match value in IP6AT

3.3.2.3.2 Flexible Filter

the 82598 supports a total of four host flexible filters. Each filter is can be configured to recognize any arbitrary pattern within the first 128 byte of the packet. To configure the flexible filter, the software programs the required values into the Flexible Host Filter Table (FHFT). These contain separate values for each filter. The software must also enable the filter in the WUFC register, and enable the overall wake up functionality must be enabled by setting PME_En in the PMCSR or the WUC register.

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 byte programmed in the Flexible Host Filter Table (FHFT) then the filter fails that packet. If the filter reaches the required length without failing the packet, it passes the packet and generates a wake-up event. It ignores any mask bits set to 1b beyond the required length.

Packets that passed the wake-up flexible filter should cause a wake-up event only if it is directed to the 82598 (passed L2 and VLAN filtering).

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 is done.

The following packets are listed for reference purposes only. The flexible filter can be used to filter these packets.

3.3.2.3.2.1 IPX Diagnostic Responder Request Packet

An IPX Diagnostic Responder Request packet must contain a valid MAC address, a Protocol Type of 0x8137, and an IPX Diagnostic Socket of 0x0456. It can also include LLC/SNAP Headers and VLAN Tags. Since filtering this packet relies on the flexible filters, which use offsets specified by the operating system directly, the operating system must account for the extra offset LLC/SNAP Headers and VLAN tags.

Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	

6	6	Source Address		Skip	
12	8	Possible LLC/SNAP Header		Skip	
12	4	Possible VLAN Tag		Compare	
12	2	Type	0x8137	Compare	IPX
14	16	Some IPX Stuff	-	Ignore	
30	2	IPX Diagnostic Socket	0x0456	Compare	

3.3.2.3.2.2 Directed IPX Packet

A valid Directed IPX packet contains the station's MAC address, a Protocol Type of 0x8137, and an IPX Node Address that equals to the station's MAC address. It can also include LLC/SNAP Headers and VLAN Tags. Since filtering this packet relies on the flexible filters, which use offsets specified by the operating system directly, the operating system must account for the extra offset LLC/SNAP Headers and VLAN tags.

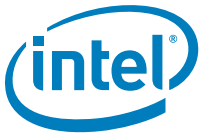
Offset	# of Bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC header – processed by main address filter
6	6	Source Address		Skip	
12	8	Possible LLC/SNAP Header		Skip	
12	4	Possible VLAN Tag		Compare	
12	2	Type	0x8137	Compare	IPX
14	10	Some IPX Stuff	-	Ignore	
24	6	IPX Node Address	Receive Address 0	Compare	Must match Receive Address 0

3.3.2.3.2.3 IPv6 Neighbor Discovery Filter

In IPv6, a neighbor discovery packet is used for address resolution. A flexible filter can be used to check for a neighbor discovery packet.

3.3.2.3.3 Wake-Up Packet Storage

The 82598 saves the first 128-byte of the wake-up packet in its internal buffer, which can be read through the WUPM register after the system wakes up.



3.4 NVM Map (EEPROM)

3.4.1 EEPROM General Map

Table 3-45 lists the EEPROM map used with 82598:

Table 3-45. EEPROM Map

Word	Used By	High Byte	Low Byte
0x00	HW	EEPROM Control Word 1	
0x01	HW	EEPROM Control Word 2	
0x02	HW	Hardware Reserved	
0x03	HW	PCIe Analog Pointer	
0x04	HW	Core 0 Configuration Pointer	
0x05	HW	Core 1 Configuration Pointer	
0x06	HW	PCIe General Pointer	
0x07	HW	PCIe Configuration 0 Pointer	
0x08	HW	PCIe Configuration 1 Pointer	
0x09	HW	Core 0 Section Pointer	
0x0A	HW	Core 1 Section Pointer	
0x0B	HW	MAC 0 Section Pointer	
0x0C	HW	MAC 1 Section Pointer	
0x0D	HW	Reserved	
0x0E	HW	Reserved	
0x0F	FW	Firmware Section Pointer	
0x10 – 0x14	SW	Compatibility High	Compatibility low
0x15	SW	PBA, Byte 1	PBA, Byte 2
0x16	SW	PBA, Byte 3	PBA, Byte 4
0x17	SW	iSCSI Boot Configuration Start Address	
0x18 – 0x2E		Software Reserved	
Word	**Used By**	**High Byte**	**Low Byte**
0x2F	SW	VPD Pointer	
0x30	PXE	PXE Word 0 (Software Use) Configuration	

Table 3-45. EEPROM Map

0x31	PXE	PXE Word 1 (Software Use) Configuration
0x32	PXE	PXE Word (Software Use) PXE Version
0x33	PXE	PXE Word (Software Use) EFI Version
0x34 – 0x37	PXE	PXE Words
0x38	HW	EEPROM Control Word 3
0x39 – 0x3E	HW	Hardware Reserved
0x3F	SW	Software Checksum, Words 0x00 – 0x3F including the areas covered by the different hardware pointers.

Table 3-45 lists the common sections for the entire EEPROM: hardware pointers, software and firmware. The hardware sections (pointed to by words 0x03 – 0x0C) are described following the common sections.

3.4.2 EEPROM Software Section

3.4.2.1 Compatibility Fields – Words 0x10-0x14

Five words in the EEPROM image are reserved for compatibility information. New bits within these fields are defined as the need arises for determining software compatibility between various hardware revisions.

3.4.2.2 PBA Number Module – Words 0x15:0x16

The nine-digit Printed Board Assembly (PBA) number used for Intel manufactured Network Interface Cards (NICs) is stored in EEPROM.

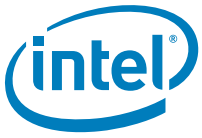
Through the course of hardware ECOs, the suffix field is incremented. The purpose of this information is to enable customer support (or any user) to identify the revision level of a product.

Network driver software should not rely on this field to identify the product or its capabilities.

PBA numbers have exceeded the length that can be stored as HEX values in two words. For newer NICs, the high word in the PBA Number Module is a flag (0xFAFA) indicating that the actual PBA is stored in a separate PBA block. The low word is a pointer to the starting word of the PBA block.

The following shows the format of the PBA Number Module field for new products.

PBA Number	Word 0x8	Word 0x9
G23456-003	FAFA	Pointer to PBA Block



The following provides the format of the PBA block; pointed to by word 0x9 above:

Word Offset	Description
0x0	Length in words of the PBA Block (default is 0x6)
0x1 ... 0x5	PBA Number stored in hexadecimal ASCII values.

The new PBA block contains the complete PBA number and includes the dash and the first digit of the 3-digit suffix which were not included previously. Each digit is represented by its hexadecimal-ASCII values.

The following shows an example PBA number (in the new style):

PBA Number	Word Offset 0	Word Offset 1	Word Offset 2	Word Offset 3	Word Offset 4	Word Offset 5
G23456-003	0006	4732	3334	3536	2D30	3033
	Specifies 6 words	G2	34	56	-0	03

Older NICs have PBA numbers starting with [A,B,C,D,E] and are stored directly in words 0x8-0x9. The dash in the PBA number is not stored; nor is the first digit of the 3-digit suffix (the first digit is always 0b for older products).

The following example shows a PBA number stored in the PBA Number Module field (in the old style):

PBA Number	Byte 1	Byte 2	Byte 3	Byte 4
E23456-003	E2	34	56	03

3.4.2.3 Software EEGEN Work Area

3.4.2.3.1 DS_Version – Word 0x29

Bits	Name	Default	Description
15:00	DS_Version	0	Dev_Starter version used as a basis for the EEPROM image.

3.4.2.3.2 OEM Version and ID – Word 0x2A

Optional identifiers that allow a user to write a version and OEM identifier in the EEPROM image.

Bits	Name	Default	Description
15:12	OEM _Version, Minor #	0	Minor # written to the middle 8 bits or Word 0x08
11:4	OEM _Version, Major #	0	Major # written to the top 4 bits of Word 0x08.
3:0	OEM ID	0	

3.4.2.3.3 Software Init Section Pointer – Word 0x2

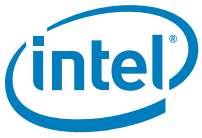
Bits	Name	Default	Description
15:00	SIS_pointer		Software Init Section pointer.

3.4.2.3.4 eTrack_ID – Word 0x2D:2E

2D	2E	Name	Description
Bits	Bits	eTrack_ID	32-bit SQL-generated number written when an image is created by EEGEN on the Intel LAN.
15:00	15:00		

VPD Pointer – Word 0x2F

Bits	Name	Default	Description
15:00	VPD_pointer		Pointer to Vital Product Data. Set by EEGEN during compile.



3.4.2.4 PXE Configuration Words – Word 0x30:3B

PXE configuration is controlled by the following Ewords.

3.4.2.4.1 Setup Options PCI Function 0 – Word 0x30

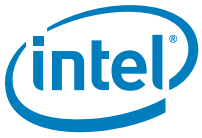
The main setup options are stored in word 30h. These options are those that can be changed by the user via the Control-S setup menu. Word 30h has the following format:

Bit(s)	Name	Function
15:13	RFU	Reserved. Must be 0.
12:10	FSD	Bits 12-10 control forcing speed and duplex during driver operation. Valid values are: 000b – Auto-negotiate 001b – 10Mbps Half Duplex 010b – 100Mbps Half Duplex 011b – Not valid (treated as 000b) 100b – 10Mbps Full Duplex 101b – 100Mbps Full Duplex 111b – 1000Mbps Full Duplex Default value is 000b.
9	RSV	Reserved. Set to 0.
8	DSM	Display Setup Message. If the bit is set to 1, the "Press Control-S" message is displayed after the title message. Default value is 1.
7-:6	PT	Prompt Time. These bits control how long the CTRL-S setup prompt message is displayed, if enabled by DIM. 00 = 2 seconds (default) 01 = 3 seconds 10 = 5 seconds 11 = 0 seconds Note: CTRL-S message is not displayed if 0 seconds prompt time is selected.
5	DEP	Deprecated. Must be 0.
4:3	DBS	Default Boot Selection. These bits select which device is the default boot device. These bits are only used if the agent detects that the BIOS does not support boot order selection or if the MODE field of word 31h is set to MODE_LEGACY. 00 = Network boot, then local boot (default) 01 = Local boot, then network boot 10 = Network boot only 11 = Local boot only
2	DEP	Deprecated. Must be 0.
1:0	PS	Protocol Select. These bits select the active boot protocol. 00 = PXE (default value) 01 = RPL (only if RPL is in the flash) 10 = iSCSI Boot primary port (only if iSCSI Boot is using this adapter) 11 = iSCSI Boot secondary port (only if iSCSI Boot is using this adapter) Only the default value of 00b should be initially programmed into the adapter; other values should only be set by configuration utilities.

3.4.2.4.2 Configuration Customization Options PCI Function 0 – Word 0x31

Word 31h of the EEPROM contains settings that can be programmed by an OEM or network administrator to customize the operation of the software. These settings cannot be changed from within the Control-S setup menu. The lower byte contains settings that would typically be configured by a network administrator using an external utility; these settings generally control which setup menu options are changeable. The upper byte is generally settings that would be used by an OEM to control the operation of the agent in a LOM environment, although there is nothing in the agent to prevent their use on a NIC implementation. The default value for this word is 4000h.

Bit(s)	Name	Function
15:14	SIG	Signature. Must be set to 01 to indicate that this word has been programmed by the agent or other configuration software.
13	RFU	Reserved. Must be 0.
12	RFU	Reserved. Must be 0.
11	RETRY	Selects Continuous Retry operation. If this bit is set, IBA will NOT transfer control back to the BIOS if it fails to boot due to a network error (such as failure to receive DHCP replies). Instead, it will restart the PXE boot process again. If this bit is set, the only way to cancel PXE boot is for the user to press ESC on the keyboard. Retry will not be attempted due to hardware conditions such as an invalid EEPROM checksum or failing to establish link. Default value is 0.
10:8	MODE	Selects the agent's boot order setup mode. This field changes the agent's default behavior in order to make it compatible with systems that do not completely support the BBS and PnP Expansion ROM standards. Valid values and their meanings are: 000b Normal behavior. The agent will attempt to detect BBS and PnP Expansion ROM support as it normally does. 001b Force Legacy mode. The agent will not attempt to detect BBS or PnP Expansion ROM supports in the BIOS and will assume the BIOS is not compliant. The user can change the BIOS boot order in the Setup Menu. 010b Force BBS mode. The agent will assume the BIOS is BBS-compliant, even though it may not be detected as such by the agent's detection code. The user can NOT change the BIOS boot order in the Setup Menu. 011b Force PnP Int18 mode. The agent will assume the BIOS allows boot order setup for PnP Expansion ROMs and will hook interrupt 18h (to inform the BIOS that the agent is a bootable device) in addition to registering as a BBS IPL device. The user can NOT change the BIOS boot order in the Setup Menu. 100b Force PnP Int19 mode. The agent will assume the BIOS allows boot order setup for PnP Expansion ROMs and will hook interrupt 19h (to inform the BIOS that the agent is a bootable device) in addition to registering as a BBS IPL device. The user can NOT change the BIOS boot order in the Setup Menu. 101b Reserved for future use. If specified, is treated as a value of 000b. 110b Reserved for future use. If specified, is treated as a value of 000b. 111b Reserved for future use. If specified, is treated as a value of 000b.
7	RFU	Reserved. Must be 0.
6	RFU	Reserved. Must be 0.
5	DFU	Disable Flash Update. If this bit is set to 1, the user is not allowed to update the flash image using PROSet. Default value is 0.
4	DLWS	Disable Legacy Wakeup Support. If this bit is set to 1, the user is not allowed to change the Legacy OS Wakeup Support menu option. Default value is 0.
3	DBS	Disable Boot Selection. If this bit is set to 1, the user is not allowed to change the boot order menu option. Default value is 0.
2	DPS	Disable Protocol Select. If set to 1, the user is not allowed to change the boot protocol. Default value is 0.



1	DTM	Disable Title Message. If this bit is set to 1, the title message displaying the version of the Boot Agent is suppressed; the Control-S message is also suppressed. This is for OEMs who do not wish the boot agent to display any messages at system boot. Default value is 0.
0	DSM	Disable Setup Menu. If this bit is set to 1, the user is not allowed to invoke the setup menu by pressing Control-S. In this case, the EEPROM may only be changed via an external program. Default value is 0.

3.4.2.4.3 PXE Version – Word 0x32

Word 32h of the EEPROM is used to store the version of the boot agent that is stored in the flash image. When the Boot Agent loads, it can check this value to determine if any first-time configuration needs to be performed. The agent then updates this word with its version. Some diagnostic tools to report the version of the Boot Agent in the flash also read this word. The format of this word is:

Bit(s)	Name	Function
15 - 12	MAJ	PXE Boot Agent Major Version. Default value is 0.
11 – 8	MIN	PXE Boot Agent Minor Version. Default value is 0.
7 – 0	BLD	PXE Boot Agent Build Number. Default value is 0.

3.4.2.4.4 IBA Capabilities – Word 0x33

Word 33h of the EEPROM is used to enumerate the boot technologies that have been programmed into the flash. This is updated by flash configuration tools and is not updated or read by IBA.

Bit(s)	Name	Function
15 - 14	SIG	Signature. Must be set to 01 to indicate that this word has been programmed by the agent or other configuration software.
13 – 5	RFU	Reserved. Must be 0.
4	ISCSI	iSCSI Boot is present in flash if set to 1.
3	EFI	EFI UNDI driver is present in flash if set to 1.
2	RPL	RPL module is present in flash if set to 1.
1	UNDI	PXE UNDI driver is present in flash if set to 1.
0	BC	PXE Base Code is present in flash if set to 1.

3.4.2.4.5 Setup Options PCI Function 1 – Word 0x34

This word is the same as word 30h, but for function 1 of the device.

3.4.2.4.6 Configuration Customization Options PCI Function 1 – Word 0x35

This word is the same as word 31h, but for function 1 of the device.

3.4.2.4.7 Setup Options PCI Function 2 – Word 0x38

This word is the same as word 30h, but for function 2 of the device.

3.4.2.4.8 Configuration Customization Options PCI Function 2 – Word 0x39

This word is the same as word 31h, but for function 2 of the device.

3.4.2.4.9 Setup Options PCI Function 3 – Word 0x3A

This word is the same as word 30h, but for function 3 of the device.

3.4.2.4.10 Configuration Customization Options PCI Function 3 – Word 0x3B

This word is the same as word 31h, but for function 3 of the device.

3.4.2.5 EEPROM Checksum Calculation

```
#define IXGBE_EEPROM_CHECKSUM     0x3F
#define IXGBE_EEPROM_SUM          0xBABA
#define IXGBE_PCIE_ANALOG_PTR     03
#define IXGBE_FW_PTR             0F
static u16 ixgbe_eeprom_calc_checksum(struct ixgbe_hw *hw)
{
    u16 i;
    u16 j;
    u16 checksum = 0;
    u16 length = 0;
    u16 pointer = 0;
    u16 word = 0;

    /* Include 0x0-0x3F in the checksum */
    for (i = 0; i < IXGBE_EEPROM_CHECKSUM; i++) {
        if (ixgbe_eeprom_read(hw, i, &word) != IXGBE_SUCCESS) {
            DEBUGOUT("EEPROM read failed\n");
            break;
        }
        checksum += word;
    }

    /* Include all data from pointers except for the fw pointer */
    for (i = IXGBE_PCIE_ANALOG_PTR; i < IXGBE_FW_PTR; i++) {
        ixgbe_eeprom_read(hw, i, &pointer);

        /* Make sure the pointer seems valid */
        if (pointer != 0xFFFF && pointer != 0) {
            ixgbe_eeprom_read(hw, pointer, &length);

            if (length != 0xFFFF && length != 0) {
                for (j = pointer+1; j <= pointer+length; j++) {
                    ixgbe_eeprom_read(hw, j, &word);
                    checksum += word;
                }
            }
        }
    }
```

```
    checksum = (u16)IXGBE_EEPROM_SUM – checksum;

    return checksum;
}
```

3.4.3 Hardware EEPROM Sections

3.4.3.1 EEPROM Init Section

The init section (EEPROM control word 0x1, 0x2, and 0x38) are read after a PE_RST_N, LAN_PWR_GOOD, or internal power on reset.

3.4.3.1.1 EEPROM Control Word 1 – Address 0x00

Bits	Name	Default	Description
15:12	Reserved	0000b	Reserved.
11:8	EEPROM Size	0010b	These bits indicate the EEPROM's actual size. Mapped to EEC[14:11]. Supported values: 0000b = Reserved. 0001b = 256 bytes. 0010b = 512 bytes. 0011b = 1 kB. 0100b = 2 kB. 0101b = 4 kB. 0110b = 8 kB. 0111b = 16 kB. 1000b = 32 kB. 1001b: 1111b = Reserved.
7:6	Signature	01b	The *Signature* field indicates to the 82598 that there is a valid EEPROM present. If the *Signature* field is not 01b, the other bits in this word are ignored, no further EEPROM read is performed, and the default values are used for the configuration space IDs.
5	MNG Enable	0b	Manageability Enable When set, indicates that the manageability block is enabled. When cleared, the manageability block is disabled (clock gated). Mapped to GRC.MNG_EN.
4	EEPROM Protection	0b	If set to 1b, all EEPROM protection schemes are enabled.
3:0	HEPSize	0000b	Hidden EEPROM Block Size This field defines the EEPROM area accessible only by manageability firmware. It can also be used to store secured data and other manageability functions. The size in bytes of the secured area equals: 0 bytes (if HEPSize equals zero), or 2^ HEPSize bytes (2 bytes, 4 bytes, …32 kB.)

3.4.3.1.2 EEPROM Control Word 2 – Address 0x01

Bits	Name	Default	Description
15:12	Reserved	0000b	Reserved.
3	Deadlock Timeout Enable	1b	If set, an 82598 that was granted access to the EEPROM that does not toggle the interface for one second has the grant revoked.
2	Core PLL Gate Disable	0b	When set, disables the gating of the core PLL in 82598 low power states.
1	Core Clocks Gate Disable	0b	When set, disables the gating of the core clock in 82598 low power states.
0	PCIe PLL Gate Disable	0b	When set, disables the gating of the PCIe PLL in L1/L2 states.

3.4.3.1.3 EEPROM Control Word 3 – Address 0x38

Bits	Name	Default	Description
15:2	Reserved	0b	Reserved.
1	APM Enable Port 1	0b	Initial value of advanced power management wake up enable in the General Receive Control register (GRC.APME). Mapped to GRC.APME of port 1.
0	APM Enable Port 0	0b	Initial value of advanced power management wake up enable in the General Receive Control register (GRC.APME). Mapped to GRC.APME of port 0.

3.4.3.2 EEPROM Hardware Pointers

Most of EEPROM words (0x0:0xF) contain hardware pointers to different hardware sections. Note that if a pointer field is 0xFFFF the appropriate section that the pointer is referring to is not present in the EEPROM.

3.4.3.2.1 Analog Configuration Sections – Words 0x04:0x05

- Port 0 Configuration Pointer
- Port 1 Configuration Pointer

These sections are loaded after LAN_PWR_GOOD or internal power on reset and contain the analog default configurations for 82598's analog parts. Words 0x4-0x5 are the pointers for these sections (The exact EEPROM address, in words).

The structure of all sections is similar as listed in the following table:

Offset	High Byte	Low Byte

	Section Length	
0x01	Configuration Data	Configuration Address

3.4.3.2.1.1 EEPROM Analog Configuration – Section Length

The section length word contains the length of the section in words. Note that the sections length does not contain the section length word itself.

3.4.3.2.1.2 EEPROM Analog Configuration – Data Word

Each word in the analog configuration section has the same structure: bits 7:0 are the register address and bits 15:8 are the registers data. The analog registers are eight bits wide with an 8-bit address width. After reading the EEPROM word, the register specified in bits 7:0 is loaded with the data from bits 15:8.

3.4.3.2.2 PCIe Analog Pointer – Word 0x03

These sections are loaded only after PE_RST_N. These sections contain the analog default configurations for 82598's analog parts. Word 0x3 is the pointer for this section (The exact EEPROM address, in words).

The structure of this section is listed in the following table:

Offset	High Byte	Low Byte
	Section Length = 0xXX	
0x1	PCIe Analog Selector 1	
0x2	PCIe Analog Word 1 – Selector 1	
0x3	PCIe Analog Word 2 – Selector 1	
0x4	PCIe Analog Word 3 – Selector 1	
0x5	PCIe Analog Selector 2	
0x6	PCIe Analog Word 1– Selector 2	
…..	……	

3.4.3.2.2.1 PCIe Analog – Section Length

The section length word contains the length of the section in words. Note that the sections length does not contain the section length word itself.

3.4.3.2.2.2 PCIe Analog Selector

Each part of the PCIe analog is reachable through a 2-byte bus (data/word). At the same time, access must signal which part of the PCIe analog it targeted (SR0, SR1, … SC). To map this access, the EEPROM uses a structure of one PCIe analog selector that indicates which internal target is accessed by the next EEPROM word.

Bit	Name	Default	Description
15:8	Selector TAG = 0xFF	0xFF	The 0xFF value in this EEPROM word identifies this word as a selector.
7:0	Target ID	0x0	Identifies which internal PCIe analog target is accessed by the following EEPROM word. Refer to the following table.

Target ID	Target
00	SR Lane 0 Registers.
01	SR Lane 1 Registers.
02	SR Lane 2 Registers.
03	SR Lane 3 Registers.
04	SR Lane 4 Registers.
05	SR Lane 5 Registers.
06	SR Lane 6 Registers.
07	SR Lane 7 Registers.
08	SR registers of all lanes at the same time.
09	SC Register.

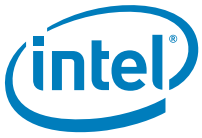
3.4.3.2.2.3 PCIe Analog Word

Bit	Name	Default	Description
15:8	Add	0x0	Register address in the PCIe ANA target.
7:0	Data	0x0	The value to write in the register pointed to by the address.

3.4.3.3 EEPROM PCIe General Configuration Section

This section is loaded after a PCIe power good or PCIe in-band reset de-assertion. It contains general configuration for the PCIe interface (not function specific) and is pointed to by word 0x06 in the EEPROM (full-byte address; must be word aligned).

Offset	High Byte	Low Byte	Section
	Section Length = 0x0B		3.4.3.3.1
0x01	PCIe Init Configuration 1		3.4.3.3.2



0x02	PCIe Init Configuration 2	3.4.3.3.4
0x03	PCIe Init Configuration 3	3.4.3.3.4
0x04	PCIe Control Offset 4	3.4.3.3.5
0x05	PCIe Control Offset 5	3.4.3.3.6
0x06	PCIe Control Offset 6	3.4.3.3.7
0x07	PCIe Control Offset 7	3.4.3.3.8
0x08	PCIe Control Offset 8	3.4.3.3.9
0x09	PCIe Control Offset 9	3.4.3.3.10
0x0A	PCIe Control Offset 10	3.4.3.3.11
0x0B	PCIe Control Offset 11	3.4.3.3.12

3.4.3.3.1 PCIe General Configuration – Section Length

The section length word contains the length of the section in words. Note that the sections length does not contain the section length word itself.

3.4.3.3.2 PCIe Init Configuration 1 – Offset 1

Bit	Name	Default	Description
15	L0s Enable	0b	If the *L0s Enable* bit is set, the default value of the *Active State Link PM Control* field in the PCIe Link Control Register is set to 01b (L0s entry enabled).
14:12	L1_Act_Ext_Latency	110b	L1 active exit latency for the configuration space Default = (32 µs-64 µs). Defined encoding: 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 = More than 64 µs. These bits configure the initial hardware value of the PCI configuration, Link CAP, and L1 Exit Latency registers following power up.
Bit	**Name**	**Default**	**Description**

11:9	L1_Act_Acc_Latency	110b	L1 active acceptable latency for the configuration space Default = (32 µs-64 µs). Defined encoding: 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 = No limit. These bits configure the initial hardware value of the PCI configuration, Device CAP, and Endpoint L1 Acceptable Latency registers following power up.
8:6	L0s_Acc_Latency	011b	L0s acceptable latency for the configuration space Default = (512 ns). Defined encoding: 000b = Less than 64 ns. 001b = 64 ns – 128 ns. 010b = 128 ns – 256 ns. 011b = 256 ns – 512 ns. 100b = 512 ns – 1 µs. 101b = 1 µs – 2 µs. 110b = 2 µs – 4 µs. 111b = No limit. These bits configure the initial hardware value of the PCI configuration, Device CAP, and Endpoint L0s Acceptable Latency registers following power up.
5:3	L0s_Se_Ext_Latency	110b	L0s exit latency for active state power management (separated reference clock) – (latency between 2 µs – 4 µs). Defined encoding: 000b = Less than 64 ns. 001b = 64 ns – 128 ns. 010b = 128 ns – 256 ns. 011b = 256 ns – 512 ns. 100b = 512 ns – 1 µs. 101b = 1 µs – 2 µs. 110b = 2 µs – 4 µs. 111b = More than 4 µs. These bits configure the initial hardware value of the PCI configuration, Link CAP, and L0s Exit Latency registers (when working with a separate clock) following power up.
2:0	L0s_Co_Ext_Latency	110b	L0s exit latency for active state power management (common reference clock) – (latency between 2 µs – 4 µs). Defined encoding: 000b = Less than 64 ns. 001b = 64 ns – 128 ns. 010b = 128 ns – 256 ns. 011b = 256 ns – 512 ns. 100b = 512 ns – 1 µs. 101b = 1 µs – 2 µs. 110b = 2 µs – 4 µs. 111b = More than 4 µs. These bits configure the initial hardware value of the PCI configuration, Link CAP, and L0s Exit Latency registers (when working with a common clock) following power up.

3.4.3.3.3 PCIe Init Configuration 2 – Offset 2

Bit	Name	Default	Description
15:12	Reserved	0b	Reserved.
11:8	Extra NFTS	0x7	Extra Number of Fast Training Signal (NFTS) that is added to the original requested number of NFTS (as requested by the upstream component).
7:0	NFTS	0xB0	Number of special sequences for L0s transition to L0. The 82598 requires at least 0xB0 NFTS for proper functionality.

3.4.3.3.4 PCIe Init Configuration 3 – Offset 3

Bit	Name	Default	Description
15	Master_Enable	0b	When set to 1b, this bit enables the PHY to be a master (upstream component/ cross link functionality).
14	Scram_dis	0b	Scrambling Disable When set to 1b, this bit disables PCIe LFSR scrambling.
13	Ack_Nak_Sch	0b	ACK/NAK Scheme 0b = Scheduled for transmission following any TLP. 1b = Scheduled for transmission according to timeouts specified in the PCIe specification.
12	Cache_Lsize	0b	Cache Line Size 0b = 64 bytes. 1b = 128 bytes.
11:10	GIO_Cap	10b	PCIe Capability Version This field must be set to 10b to use extended configuration capability (used for a timeout mechanism). This bit is mapped to GCR.PCIe_Capability_Version.
9	IO_Sup	1b	I/O Support (effects I/O BAR request). When set to 1b, I/O is supported.
8	Packet_Size	1b	Default Packet Size 0b = 128 bytes. 1b = 256 bytes.
7:6	Lane_Width	11b	Max Link Width 00b = 1 lane. 01b = 2 lanes. 10b = 4 lanes. 11b = 8 lanes.
5	Elastic Buffer Diff1	0b	When set to 1b, the elastic buffers are activated in a more limited mode (read and write pointers distance wise).
4	Elastic buffer ctrl	0b	When set to 1b, sets the elastic buffers to a mode that sets phase-only mode during electrical-idle states.
3:2	Act_Stat_PM_Sup	11b	Determines support for Active State Link Power Management (ASLPM). Loaded into the PCIe Active State Link PM Support register.

Bit	Name	Default	Description
1	Slot_Clock_Cfg	1b	When set, 82598 uses the PCIe reference clock supplied on the connector (for add-in solutions).
0	Loop Back Polarity Inversion	0b	Checks the polarity inversion in a loop-back master entry.

3.4.3.3.5 PCIe Control – Offset 4

Bit	Name	Default	Description
15	Reserved	0b	Reserved.
14	DLLP Timer Enable	0b	When set, enables the DLLP timer counter.
13	GIO Down Reset Disable	0b	Disables a core reset when the PCIe link goes down.
12	Lane Reversal Disable	0b	Disables the ability to negotiate a lane reversal.
11	Good Recovery	0b	When this bit is set, the LTSSM recovery states always progress towards linkup (force a good recovery when recovery occurs).
10	Leaky Bucket Disable	1b	Disables the leaky bucket mechanism in the PCIe PHY. Disabling this mechanism holds the link from going to a recovery retrain in case of disparity errors.
9:7	Reserved	0b	Reserved.
6	GIO TS Retrain Mode	0b	Controls the condition of an LTSSM entry to recovery.
5	L2 Disable	0b	Disables the link from entering the L2 state.
4	Skip Disable	0b	Disables the skip symbol insertion in the elastic buffer.
3	Reserved	0b	Reserved.
2	Electrical Idle	0b	Electrical Idle Mask If set to 1b, disables a check for an illegal electrical idle sequence (for example, eidle ordered set without common mode and vise versa) and excepts any of them as the correct eidle sequence. **Note:** The specification can be interpreted so that the eidle ordered set is sufficient for a transition to any of the power management states. The use of this bit enables such interpretation and avoids the possibility of correct behavior being understood as illegal sequences.
1:0	Latency_To_Enter_L1	11b	Period (in the L0s state) before transitioning into an L1 state. 00b = 64 µs. 01b = 256 µs. 10b = 1 ms. 11b = 4 ms.

3.4.3.3.6 PCIe Control – Offset 5

Bit	Name	Default	Description
15:11	Reserved	0b	Reserved.
10	LAN Function Select	0b	When the *LAN Function Select* field = 0b, LAN 0 is routed to PCI function 0 and LAN 1 is routed to PCI function 1. If the *LAN Function Select* field = 1b, LAN 0 is routed to PCI function 1 and LAN 1 is routed to PCI function 0. This bit is mapped to FACTPS[30].
9:8	Reserved	0b	Reserved.
7	Completion Timeout Disable	0b	Disables the PCIe completion timeout mechanism. This bit is mapped to GCR.Completion_Timeout_Disable. 0b = Completion timeout enabled. 1b = Completion time out disabled.
6:5	Completion Timeout Value	00b	Determines the range of the PCIe completion timeout. 00b = 0.5 ms to 1 ms. 01b = 50 ms to 100 ms. 10b – 0.5 s to 1 s. 11b = 10 s to 20 s.
4	Completion Timeout Resend	1b	When set, enables a response to a request once the completion timeout expired. This bit is mapped to GCR.Completion_Timeout_Resend. 0b = Do not resend request on completion timeout. 1b = Resend request on completion timeout.
3	Dummy Function Enable	0b	Enables support for dummy function when only Function is needed. 0b = Disable. 1b = Enable.
2	Wake_pin_enable	1b	Enables the use of the wake pin for a PME event in all power states.
1	LAN PCI Disable	0b	LAN PCI Disable When set to 1b, one LAN port is disabled and the appropriate PCI function might be disabled and act as a dummy function. The function that is disabled is determined by the *LAN PCI Function Select* field. If the disabled function is function 0, it acts as a dummy function.
0	LAN Disable Select	0b	LAN Disable Select 0b = LAN 0 is disabled. 1b = LAN 1 is disabled.

3.4.3.3.7 PCIe Control – Offset 6 – LAN Power Consumption

Bit	Name	Default	Description
15:8	LAN D0 Power	0x0	The value in this field is reflected in the PCI Power Management Data register of the LAN functions for D0 power consumption and dissipation (*Data_Select* = 0 or 4). Power is defined in 100 mW units. The power includes also the external logic required for the LAN function.
7:5	Function 0 Common Power	0x0	The value in this field is reflected in the PCI Power Management Data register of function 0 when the *Data_Select* field is set to 8 (common function). The MSBs in the Data register that reflects the power values are padded with zeros. When one port is used, this field should be set to 0b.
4:0	LAN D3 Power	0x0	The value in this field is reflected in the PCI Power Management Data register of the LAN functions for D3 power consumption and dissipation (*Data_Select* = 3 or 7). Power is defined in 100 mW units. The power includes also the external logic required for the LAN function. The MSBs in the Data register that reflects the power values are padded with zeros.

3.4.3.3.8 PCIe Control – Offset 7

Bit	Name	Default	Description
15:11	Reserved	0x0	Reserved.
10:8	Flash Size	000b	Indicates Flash Size 000b = 64 kB. 001b = 128 kB. 010b = 256 kB. 011b = 512 kB. 100b = 1 MB. 101b = 2 MB. 110b = 4 MB. 111b = 8 MB. The Flash size impacts the requested memory space for the Flash and expansion ROM BARs in PCIe configuration space.
7	Reserved	0b	Reserved.
6:2	Go Electrical Idle Delay	0b	Permits a tune delay between the electrical idle symbol sent on the physical lane and the go-electrical idle command to GIO-ana.
1	Load Subsystem IDs	1b	When set to 1b, indicates that the function is to load its PCIe sub-system ID and sub-system vendor ID from the EEPROM (offset 0x8 and 0x9 in this section).
0	Load Device ID	1b	When set to 1b, indicates that the function is to load its PCI device ID from the EEPROM (offset 10 in this section and offset 2 in PCIe configuration space 0/1 section).

3.4.3.3.9 PCIe Control – Offset 8 – Sub-System ID

If the load sub-system IDs in offset 0x7 of this section is set, this word is read in to initialize the sub-system ID. The default value is 0x0.

3.4.3.3.10 PCIe Control – Offset 9 – Sub-System Vendor ID

If the load sub-system IDs in offset 0x7 of this section is set, this word is read in to initialize the sub-system vendor ID. The default value is 0x8086.

3.4.3.3.11 PCIe Control – Offset 10 – Dummy Device ID

If the load vendor/device IDs in offset 0x7 of this section is set, this word is read in to initialize the device ID of the dummy device in this function (if enabled). The default value is 0x10A6.

3.4.3.3.12 PCIe Control – Offset 11 – Device Revision ID

Bit	Name	Default	Description
7:0	DEVREVID	0x0	Device Rev ID The actual device revision ID is the EEPROM value XORed with the hardware value (0x00 for 82598 A-0).

3.4.3.4 EEPROM PCIe Configuration Space 0/1 Sections

Word 0x7 points on the PCIe configuration space defaults of function 0 while word 0x8 points to function 1 defaults. Both sections are loaded at the de-assertion of LAN_PWR_GOOD, internal power on reset, and a PCIe in-band reset. In addition, function 0 defaults are loaded after the de-assertion of PCIe config 0 reset; function 1 defaults are loaded after the de-assertion of PCIe config 1 reset.

The structure of both sections is identical as listed in the following table:

Offset	High Byte	Low Byte	Section
	Section Length = 0x2		3.4.3.4.1
0x1	PCIe Configuration Space 0/1 Control 1		3.4.3.4.2
0x2	PCIe Configuration Space 0/1 Control 2		3.4.3.4.3

3.4.3.4.1 PCIe Configuration Space 0/1 – Section Length

The section length word contains the length of the section in words. Note that the sections length does not contain the section length word itself.

3.4.3.4.2 EEPROM PCIe Configuration Space 0/1- Offset 1

Bit	Name	Default	Description
15:10	Reserved	0x0	Reserved.
9	LAN Flash Disable	1b	A value of 1b disables the Flash logic. The Flash access BAR in the PCI configuration space is also disabled.
8	LAN Boot Disable	1b	A value of 1b disables the expansion ROM BAR in the PCI configuration space.
7	Interrupt Pin	0b for LAN0 1b for LAN1	Controls the value advertised in the *Interrupt Pin* field of the PCI configuration header for this device/function. A value of 0b, reflected in the *Interrupt Pin*, field indicates that 82598 uses INTA#; a value of 1b indicates that 82598 uses INTB#. When one port is used, the configuration should be aligned to make sure INTA# is used.
6:5	Reserved	00b	Reserved.
4:0	MSI_X _N	0x10	This field specifies the number of entries in the MSI-X tables for this function. MSI_X_N is equal to the number of entries minus one. For example, a return value of 0x7 means eight vectors are available. The 82598 supports a maximum of 16 vectors.

The following table lists the different combinations for bits 9 and 8.

Bit 9 (Flash Disable)	Bit 8 (Boot Disable)	Functionality (Active Windows)
0b	0b	Flash and expansion ROM BARs are active.
0b	1b	Flash BAR is enabled and expansion ROM BAR is disabled.
1b	0b	Reserved
1b	1b	Flash and Expansion ROM BARs are disabled.

3.4.3.4.3 EEPROM PCIe Configuration Space 0/1 – Offset 2 Device ID

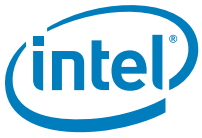
If the load vendor/device IDs in offset 0x7 of section PCIe General configuration is set, this word is read in to initialize the device ID of the LAN function. The default value is 10B6.

3.4.3.5 EEPROM Core 0/1 Section

Word 0x9 points to the core configuration defaults of core 0 while word 0xA points to core 1 defaults. Both sections are loaded at the de-assertion of their core master reset.

The structure of both sections is identical as listed in the following table:

Offset	High Byte	Low Byte	Section
	Section Length = 0x7		3.4.3.5.1
0x1	Ethernet Address Byte 2	Ethernet Address Byte 1	3.4.3.5.2



0x2	Ethernet Address Byte 4	Ethernet Address Byte 3	3.4.3.5.2
0x3	Ethernet Address Byte 6	Ethernet Address Byte 5	3.4.3.5.2
0x4	LED 1 configuration	Led 0 Configuration	3.4.3.5.3
0x5	LED 3 Configuration	Led 2 Configuration	3.4.3.5.3
0x6	SDP Control		3.4.3.5.4
0x7	Filter Control		3.4.3.5.5

3.4.3.5.1 Core Configuration Section – Section Length

The section length word contains the length of the section in words. Note that the sections length does not contain the section length word itself.

3.4.3.5.2 Ethernet Address – Offset 1-3

The Ethernet Individual Address (IA) is a 6-byte field that must be unique for each NIC and must also be unique for each copy of the EEPROM image. The first three bytes are vendor specific. For example, the IA is equal to [00 AA 00] or [00 A0 C9] for Intel products. The value of this field is loaded into the Receive Address register 0 (RAL0/RAH0).

For the purpose of this datasheet, the IA byte numbering convention is as follows:

Vendor	1	2	3	4	5	6
Intel original	00	AA	00	Variable	Variable	Variable
Intel new	00	A0	C9	Variable	Variable	Variable

3.4.3.5.3 LEDs Configuration – Offset 4-5

The LEDCTL register defaults are loaded from these two words as listed in the following table:

LED	LEDCTL Bits	EPROM Byte
LED 0 Control	7:0	Word 0x4 (low byte).
LED 1 Control	15:8	Word 0x4 (high byte).
LED 2 Control	23:16	Word 0x5 (low byte).
LED 3 Control	31:241	Word 0x5 (high byte).

Note: The content of the EEPROM words is similar to the register content.

3.4.3.5.4 SDP Control – Offset 6

Bit	Name	Default	Description
15:12	Reserved	0000b	Should be set to 0000b.
11	SDPDIR[3]	0b	SDP3 Pin – Initial Direction Mapped to ESDP.SDP3_IODIR. This bit configures the initial hardware value of the SDP3_IODIR bit in the ESDP register following power up.
10	SDPDIR[2]	0b	SDP2 Pin – Initial Direction Mapped to ESDP.SDP2_IODIR. This bit configures the initial hardware value of the SDP2_IODIR bit in the ESDP register following power up.
9	SDPDIR[1]	0b	SDP1 Pin – Initial Direction Mapped to ESDP.SDP1_IODIR. This bit configures the initial hardware value of the SDP1_IODIR bit in the ESDP register following power up.
8	SDPDIR[0]	0b	SDP0 Pin – Initial Direction Mapped to ESDP.SDP0_IODIR. This bit configures the initial hardware value of the SDP0_IODIR bit in the ESDP register following power up.
7:4	Reserved	0x0	Reserved.
3	SDPVAL[3]	0b	SDP3 Pin – Initial Output Value Mapped to ESDP.SDP3_DATA. This bit configures the initial power on value output of SDP3 (when configured as an output) by configuring the initial hardware value of the SDP3_DATA bit in the ESDP register following power up.
2	SDPVAL[2]	0b	SDP2 Pin – Initial Output Value Mapped to ESDP.SDP2_DATA. This bit configures the initial power on value output of SDP2 (when configured as an output) by configuring the initial hardware value of the SDP2_DATA bit in the ESDP register following power up.

Bit	Name	Default	Description
1	SDPVAL[1]	0b	SDP1 Pin – Initial Output Value Mapped to ESDP.SDP1_DATA. This bit configures the initial power on value output of SDP1 (when configured as an output) by configuring the initial hardware value of the SDP1_DATA bit in the ESDP register following power up.
0	SDPVAL[0]	0b	SDP0 Pin – Initial Output Value Mapped to ESDP.SDP0_DATA. This bit configures the initial power on value output of SDP0 (when configured as an output) by configuring the initial hardware value of the SDP0_DATA bit in the ESDP register following power up.

3.4.3.5.5 Filter Control – Offset 7

Bit	Name	Default	Description
15:1	Reserved	0b	Reserved.
0	ADVD3WUC	1b	D3Cold WakeUp Capability Advertisement Enable When set, D3Cold wakeup capability is advertised based on whether or not AUX_PWR advertises the presence of auxiliary power.

3.4.3.6 EEPROM MAC 0/1 Sections

Word 0xB points to the LAN MAC configuration defaults of function 0 while word 0xC points to function 1 defaults. Both sections are loaded at the de-assertion of their core master reset.

The structure of both sections is identical as listed in the following table:

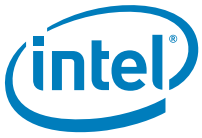
Offset	High Byte	Low Byte	Section
	Section Length = 0x5		3.4.3.6.1
0x1	Link Mode Configuration		3.4.3.6.2
0x2	Swap Configuration		3.4.3.6.3
0x3	Swizzle and Polarity Configuration		3.4.3.6.4
0x4	Auto Negotiation Default Bits		3.4.3.6.5
0x5	AUTOC2 Upper Half		3.4.3.6.6

3.4.3.6.1 MAC Configuration Section – Section Length

The section length word contains the length of the section in words. Note that the sections length does not contain the section length word itself.

3.4.3.6.2 Link Mode Configuration – Offset 1

Bit	Name	Default	Description
15:13	Link Mode Select	001b	Selects the active link mode. 000b = 1 Gb/s link (no auto negotiation). 001b = 10 Gb/s link (no auto negotiation). 010b = 1 Gb/s link with clause 37 auto negotiation enable. 011b = Reserved. 100b = KX4/KX auto negotiation enable. 1 Gb/s (Clause 37) auto negotiation disable. 101b = Reserved. 110b = KX4/KX auto negotiation enable. 1 Gb/s (Clause 37) auto negotiation enable. 111b = Reserved. These bits are mapped to AUTOC.LMS.
12	Restart AN	0b	Restarts the KX/KX4 auto negotiation process (self-clearing bit). 0b = No action needed. 1b = Restart KX/KX4 Auto Negotiation. This bit is mapped to AUTOC.Restart_Auto Negotiation.
11	RATD	0b	Restarts auto negotiation on a transition to Dx. This bit is loaded to AUTOC.RATD and mapped to AUTOC.RATD.
10	D10GMP	0b	Disables 10 Gb/s (KX4) on Dx(Dr/D3) without main power. This bit is loaded to AUTOC.D10ODG and mapped to AUTOC.D10GMP.
9	1G PMA_PMD	1b	PMA/PMD used for 1 Gb/s. 0b = BX PMA/PMD. 1b = KX PMA/PMD. This bit is mapped to AUTOC.1G_PMA_PMD.
8:7	10G PMA_PMD	00b	PMA/PMD used for 10 Gb/s. 00b = XAUI PMA/PMD. 01b = KX4 PMA/PMD. 10b = CX4 PMA/PMD. 11b = Reserved. These bits are mapped to AUTOC.10G_PMA_PMD.
6:2	ANSF	00001b	AN Selector Field This value is used as the selector field in the link control word during the clause 73 auto-negotiation process (default value is according to 802.3ap, draft 2.4). Mapped to AUTOC.ANSF.
1	ANACK2	0b	AN ACK2 Field This value is transmitted in the *Acknowledge2* field of the null next page that is transmitted during a next page handshake. Mapped to AUTOC.ANACK2.
0	Reserved	0b	Reserved.

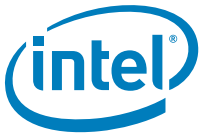


3.4.3.6.3 SWAP Configuration – Offset 2

Bit	Name	Default	Description
15:14	Swap_Rx_Lane_0	00b	Determines which port lane is mapped to MAC Rx lane 0. 00b = Port rx lane 0 to MAC Rx lane 0. 01b = Port rx lane 1 to MAC Rx lane 0. 10b = Port rx lane 2 to MAC Rx lane 0. 11b = Port rx lane 3 to MAC Rx lane 0. Mapped to SERDESC.swap_rx_lane_0.
13:12	Swap_Rx_Lane_1	01b	Determines which port lane is mapped to MAC Rx lane 1. Mapped to SERDESC.swap_rx_lane_1.
11:10	Swap_Rx_Lane_2	10b	Determines which port lane is mapped to MAC Rx lane 2. Mapped to SERDESC.swap_rx_lane_2.
9:8	Swap_Rx_Lane_3	11b	Determines which port lane is mapped to MAC Rx lane 3. Mapped to SERDESC.swap_rx_lane_3.
7:6	Swap_Tx_Lane_0	00b	Determines the port destination tx lane for MAC Tx lane 0. 00b = MAC tx lane 0 to port Tx lane 0. 01b = MAC tx lane 0 to port Tx lane 1. 10b = MAC tx lane 0 to port Tx lane 2. 11b = MAC tx lane 0 to port Tx lane 3. Mapped to SERDESC.swap_tx_lane_0.
5:4	Swap_Tx_Lane_1	01b	Determines the port destination tx lane for MAC Tx lane 1. Mapped to SERDESC.swap_tx_lane_1.
3:2	Swap_Tx_Lane_2	10b	Determines the port destination tx lane for MAC Tx lane 2. Mapped to SERDESC.swap_tx_lane_2.
1:0	Swap_Tx_Lane_3	11b	Determines the port destination tx lane for MAC Tx lane 3. Mapped to SERDESC.swap_tx_lane_3.

3.4.3.6.4 Swizzle and Polarity Configuration – Offset 3

Bit	Name	Default	Description
15:12	Swizzle_Rx	0x0	Swizzle_Rx[0] – Swizzles the bits of MAC Rx lane 0. Swizzle_Rx[1] – Swizzles the bits of MAC Rx lane 1. Swizzle_Rx[2] – Swizzles the bits of MAC Rx lane 2. Swizzle_Rx[3] – Swizzles the bits of MAC Rx lane 3. Swizzles the bits if set to 1b. Mapped to SERDESC.Swizzle_Rx_lanes.
11:8	Swizzle_Tx	0x0	Swizzle_Tx[0] – Swizzles the bits of MAC Tx lane 0. Swizzle_Tx[1] – Swizzles the bits of MAC Tx lane 1. Swizzle_Tx[2] – Swizzles the bits of MAC Tx lane 2. Swizzle_Tx[3] – Swizzles the bits of MAC Tx lane 3. Swizzles the bits if set to 1b. Mapped to SERDESC.Swizzle_Tx_lanes.
7:4	Polarity_Rx	0x0	Polarity_Rx[0] – Changes the bit polarity of MAC Rx lane 0 Polarity_Rx[1] – Changes the bit polarity of MAC Rx lane 1 Polarity_Rx[2] – Changes the bit polarity of MAC Rx lane 2 Polarity_Rx[3] – Changes the bit polarity of MAC Rx lane 3 Changes bit polarity if set to 1b. Mapped to SERDESC.Rx_lanes_polarity.
3:0	Polarity_Tx	0x0	Polarity_Tx[0] – Changes the bit polarity of MAC Tx lane 0. Polarity_Tx[1] – Changes the bit polarity of MAC Tx lane 1. Polarity_Tx[2] – Changes the bit polarity of MAC Tx lane 2. Polarity_Tx[3] – Changes the bit polarity of MAC Tx lane 3. Changes bit polarity if set to 1b. Mapped to SERDESC.Tx_lanes_polarity.



3.4.3.6.5 Auto Negotiation Defaults – Offset 4

Bit	Name	Default	Description
15:14	KX Support	1b	The value of these EEPROM settings are shown in bits A0:A1 of the *Technology Ability* field of the auto negotiation word. 00b = Illegal value. 01b = A0 = 1b, A1 = 0b. KX supported. KX4 not supported. 10b = A0 = 0b, A1 = 1b. KX not supported. KX4 supported. 11b = A0 = 1b, A1 = 1b. KX supported. KX4 supported. Mapped to AUTOC.KX_support.
13:12	Pause Bits	0x00	The value of these bits is loaded to bits D11:D10 of the link code word (pause data). Bit 12 is loaded to D11. Mapped to AUTOC.PB.
11	RF	0b	This bit is loaded to the RF of the auto negotiation word. Mapped to AUTOC.RF.
10:9	AN Parallel Detect Timer	00b	Configures the parallel detect counters. 00b = 1 ms. 01b = 2 ms. 10b = 5 ms. 11b = 8 ms. Mapped to AUTOC.ANPDT.
8	AN RX Loose Mode	1b	Enables less restricted functionality (allow 9/11 bit symbols). 0b = Disables loose mode. 1b = Enables loose mode. Mapped to AUTOC.ANRXLM.
7	AN RX Drift Mode	1b	Enables the drift caused by PPM in the RX data. 0b = Disables drift mode. 1b = Enables drift mode. Mapped to AUTOC.ANRXDM.
6:2	AN RX Align Threshold	00011b	Sets the threshold to determine that the alignment is stable. Sets how many stable symbols to find before declaring the AN_RX 10b symbol stable. Mapped to AUTOC.ANRXAT.
1:0	Reserved	00b	Reserved.

3.4.3.6.6 AUTOC2 – Upper Half– Offset 5

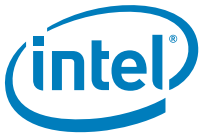
Bit	Name	Default	Description
15	Reserved	0b	Reserved.
14	Parallel Detect Disable	0b	Disables the parallel detect part in KX/KX4 auto negotiation. When set to 1b, the auto negotiation process avoids any parallel detect activity and relies only on the DME pages receive and transmit. 1b = Reserved. 0b = Enable the parallel detect (normal operation). Mapped to AUTOC2.PDD.
13:8	Reserved	0x0	Reserved
7	Latch High 10G Aligned Indication	0b	Override any deskew alignment failures in the 10 Gb/s link (by latching high). This keeps the link up after the first time it reached the AN_GOOD state in 10 Gb/s (unless RestartAN is set). Mapped to AUTOC2.LH1GAI.
6	Reserved	0b	Reserved.
5:4	AN Advertisement Page Override	00b	Override the auto negotiation advertisement page with data from the AUTOC3 register. 00b = Use internal values (default). Mapped to AUTOC2.ANAPO.
3:0	Reserved	0x0	Reserved.

3.4.4 Hardware Section – Auto-Read

The following table lists the different sections of auto-read events.

Table 3-46. EEPROM Section Auto-Read

Function	Internal Power On Reset or LAN_PWR_GOOD	PE_RST_N	PCIe Inband Reset	Function 0 D3 to D0	SW Reset 0	Link Reset 0	Function 1 D3 to D0	SW Reset 1	Link Reset 1
PCIe Analog Configuration		X							
Core 0 Configuration	X								
Core 1 Configuration	X								
PCIe General Configuration		X							
PCIe Function 0 Config Space		X	X	X					
PCIe Function 1 Config Space		X	X				X		



Function	Internal Power On Reset or LAN_PWR_GOOD	PE_RST_N	PCIe Inband Reset	Function 0 D3 to D0	SW Reset 0	Link Reset 0	Function 1 D3 to D0	SW Reset 1	Link Reset 1
Core 0 Configuration		X	X	X	X	X			
Core 1 Configuration		X	X				X	X	X
MAC Core 0 Configuration		X[1]	X[1]	X[1]	X[1]	X			
MAC Core 1 Configuration		X[1]	X[1]				X[1]	X[1]	X

1. The core and MAC core configuration sections are auto-read after the appropriate event only if the manageability unit is disabled.

3.4.5 Manageability Control Sections

The following table lists the EEPROM global offsets that are used by 82598 firmware:

Global Offset	Description	Section
0x0	Test Configuration Pointer	3.4.5.1.1
0x1	Loader Patch Pointer	3.4.5.1.2
0x2	No Manageability Patch Pointer	3.4.5.1.3
0x3	Manageability Capability/Manageability Enable	3.4.5.1.3.1
0x4	Pass Through Patch Configuration Pointer	3.4.5.1.6.1.1
0x5	Pass Through LAN 0 Configuration Pointer	3.4.5.1.6.1.2
0x6	Sideband Configuration Pointer	3.4.5.1.6.1.3
0x7	Flexible TCO Filter Configuration Pointer	3.4.5.1.6.1.4
0x8	Pass Through LAN 1 Configuration Pointer	3.4.5.1.6.1.5
0x9	NC-SI Microcode Download Pointer	3.4.5.1.6.1.6
0xA	NC-SI Configuration Pointer	

3.4.5.1 Common Firmware Pointers

Word 0x0F is used to point to firmware structures.

3.4.5.1.1 Test Configuration Pointer – (Global Offset 0x0)

Bit	Name	Description
15:0	Pointer	Pointer to test configuration structure.

3.4.5.1.2 Loader Patch Pointer – (Global Offset 0x1)

Bit	Name	Description
15:0	Pointer	Pointer to loader patch structure.

3.4.5.1.3 No Manageability Patch Pointer – (Global Offset 0x2)

Bit	Name	Description
15:0	Pointer	Pointer to no manageability patch structure.

3.4.5.1.3.1 Manageability Capability/Manageability Enable – (Global Offset 0x3)

Bit	Name	Description
15	Reserved	Reserved
14	Sideband Interface	0b = SMBus. 1b = NC-SI.
13:11	Reserved	Reserved.
10:8	Manageability Mode	0x0 = None. 0x1 = Reserved. 0x2 = PT mode. 0x3:0x7 = Reserved.
7	Port1 Manageability Cable	0b = Reserved. 1b = bits 3:0 are applicable to port 1.
6	Port0 Manageability Cable	0b = Reserved. 1b = Bits 3:0 are applicable to port 0.
5	LAN1 Force TCO Reset Disable	0b = Enable Force TCO reset on LAN1. 1b = Disable Force TCO reset on LAN1.
4	LAN0 Force TCO Reset Disable	0b = Enable Force TCO reset on LAN0. 1b = Disable Force TCO reset on LAN0.

Bit	Name	Description
3	Pass Through Cable	0b = Disable. 1b = Enable.
2:0	Reserved	Reserved

3.4.5.1.3.2 NC-SI Configuration Pointer – (Global Offset 0x4)

Bit	Name	Description
15:0	Pointer	Pointer to NC-SI configuration pointer structure.

3.4.5.1.4 Test Structure

3.4.5.1.4.1 Section Header – (Offset 0x0)

Bit	Name	Description
15:8	Block CRC	
7:0	Block Length (words)	

3.4.5.1.4.2 Loopback Test Configuration – (Offset 0x1)

Bit	Name	Description
15:2	Reserved	
1	Loopback Test Use SDP Output	
0	Loopback Test Enable	

3.4.5.1.5 Patch Structure

This structure is used for all the patches in different modes (loader, no manageability and pass through).

3.4.5.1.5.1 Patch Data Size – (Offset 0x0)

Bit	Name	Description
15:0	Data Size (bytes)	

3.4.5.1.5.2 Block CRC8– (Offset 0x1)

Bit	Name	Description
15:8	Reserved	
7:0	CRC8	

3.4.5.1.5.3 Patch Entry Point Pointer Low Word – (Offset 0x2)

Bit	Name	Description
15:0	Patch Entry Point Pointer (low word)	

3.4.5.1.5.4 Patch Entry Point Pointer High Word – (Offset 0x3)

Bit	Name	Description
15:0	Patch Entry Point Pointer (high word)	

3.4.5.1.5.5 Patch Version 1 Word – (Offset 0x4)

Bit	Name	Description
15:8	Patch Generation Hour	
7:0	Patch Generation Minutes	

3.4.5.1.5.6 Patch Version 2 Word – (Offset 0x5)

Bit	Name	Description
15:8	Patch Generation Month	
7:0	Patch Generation Day	

3.4.5.1.5.7 Patch Version 3 Word – (Offset 0x6)

Bit	Name	Description
15:8	Patch Silicon Version Compatibility	
7:0	Patch Generation Year	

3.4.5.1.5.8 Patch Version 4 Word – (Offset 0x7)

Bit	Name	Description
15:8	Patch Major Number	
7:0	Patch Minor Number	

3.4.5.1.5.9 Patch Data Words – (Offset 0x8 – Block Length)

Bit	Name	Description
15:0	Patch Firmware Data	

3.4.5.1.6 Pass Through Control Words

The following section describes the pointers and structures dedicated to pass through mode.

3.4.5.1.6.1 Pass Through Pointers

3.4.5.1.6.1.1 Pass Through Patch Configuration Pointer – (Global Offset 0x4)

Bit	Name	Description
15:0	Pointer	Pointer to pass through patch configuration pointer structure.

3.4.5.1.6.1.2 Pass Through LAN 0 Configuration Pointer – (Global Offset 0x5)

Bit	Name	Description
15:0	Pointer	Pointer to pass through LAN 0 configuration pointer structure.

3.4.5.1.6.1.3 Sideband Configuration Pointer – (Global Offset 0x6)

Bit	Name	Description
15:0	Pointer	Pointer to Sideband configuration structure.

3.4.5.1.6.1.4 Flexible TCO Filter Configuration Pointer – (Global Offset 0x7)

Bit	Name	Description
15:0	Pointer	Pointer to Flexible TCO filter configuration pointer structure.

3.4.5.1.6.1.5 Pass Through LAN 1 Configuration Pointer – (Global Offset 0x8)

Bit	Name	Description
15:0	Pointer	Pointer to pass through LAN 1 configuration pointer structure.

3.4.5.1.6.1.6 NC-SI Microcode Download Pointer – (Global Offset 0x9)

Bit	Name	Description
15:0	Pointer	Pointer to NC-SI microcode download pointer structure.

3.4.5.1.6.2 Pass Through LAN Configuration Structure

3.4.5.1.6.2.1 Section Header – (0ffset 0x0)

Bit	Name	Description
15:8	Block CRC8	
7:0	Block Length	

3.4.5.1.6.2.2 LAN 0 IPv4 Address 0 (LSB) MIPAF0 – (0ffset 0x01)

Bit	Name	Description
15:8	LAN 0 IPv4 Address 0, Byte 1	
7:0	LAN 0 IPv4 Address 0, Byte 0	

3.4.5.1.6.2.3 LAN 0 IPv4 Address 0 (MSB) (MIPAF0) – (0ffset 0x02)

Bit	Name	Description
15:8	LAN 0 IPv4 Address 0, Byte 3	
7:0	LAN 0 IPv4 Address 0, Byte 2	

3.4.5.1.6.2.4 LAN 0 IPv4 Address 1 MIPAF1 – (0ffset 0x03-x004)

Same structure as LAN0 IPv4 Address 0.

3.4.5.1.6.2.5 LAN 0 IPv4 Address 2 MIPAF2 – (0ffset 0x05-0x06)

Same structure as LAN0 IPv4 Address 0.

3.4.5.1.6.2.6 LAN 0 IPv4 Address 3 MIPAF3 – (0ffset 0x07-0x08)

Same structure as LAN0 IPv4 Address 0.

3.4.5.1.6.2.7 LAN 0 MAC Address 0 (LSB) MMAL0 – (0ffset 0x09)

Bit	Name	Description
15:8	LAN 0 MAC Address 0, Byte 1	
7:0	LAN 0 MAC Address 0, Byte 0	

3.4.5.1.6.2.8 LAN 0 MAC Address 0 (LSB) MMAL0 – (0ffset 0x0A)

Bit	Name	Description
15:8	LAN 0 MAC Address 0, Byte 3	
7:0	LAN 0 MAC Address 0, Byte 2	

3.4.5.1.6.2.9 LAN 0 MAC Address 0 (MSB) MMAH0 – (0ffset 0x0B)

Bit	Name	Description
15:8	LAN 0 MAC Address 0, Byte 5	
7:0	LAN 0 MAC Address 0, Byte 4	

3.4.5.1.6.2.10 LAN 0 MAC Address 1 MMAL/H1 – (0ffset 0x0C-0x0E)

Same structure as LAN0 MAC Address 0.

3.4.5.1.6.2.11 LAN 0 MAC Address 2 MMAL/H2 – (0ffset 0x0F-0x11)

Same structure as LAN0 MAC Address 0.

3.4.5.1.6.2.12 LAN 0 MAC Address 3 MMAL/H3 – (0ffset 0x12-0x14)

Same structure as LAN0 MAC Address 0.

3.4.5.1.6.2.13 LAN 0 UDP Flexible Filter Ports 0 – 15 (MFUTP Registers) – (0ffset 0x15-0x24)

Bit	Name	Description
15:0	LAN UDP Flexible Filter Value	

3.4.5.1.6.2.14 LAN 0 VLAN Filter 0 – 7 (MAVTV Registers) – (0ffset 0x25 – 0x2C)

Bit	Name	Description
15:12	Reserved	
11:0	LAN 0 VLAN Filter Value	

3.4.5.1.6.2.15 LAN 0 Manageability Filters Valid (MFVAL LSB) – (0ffset 0x2D)

Bit	Name	Description
15:8	VLAN	Indicates if the VLAN filter registers (MAVTV) contain valid VLAN tags. Bit 8 corresponds to filter 0, etc.
7:4	Reserved	Reserved.
3:0	MAC	Indicates if the MAC unicast filter registers (MMAH, MMAL) contain valid MAC addresses. Bit 0 corresponds to filter 0, etc.

3.4.5.1.6.2.16 LAN 0 Manageability Filters Valid (MFVAL MSB) – (0ffset 0x2E)

Bit	Name	Description
15:12	Reserved	Reserved.
11:8	IPv6	Indicates if the IPv6 address filter registers (MIPAF) contain valid IPv6 addresses. Bit 8 corresponds to address 0, etc. Bit 11 (filter 3) applies only when IPv4 address filters are not enabled (MANC.EN_IPv4_FILTER=0).
7:4	Reserved	Reserved.
3:0	IPv4	Indicates if the IPv4 address filters (MIPAF) contain a valid IPv4 address. These bits apply only when IPv4 address filters are enabled (MANC.EN_IPv4_FILTER=1).

3.4.5.1.6.2.17 LAN 0 MANC Value LSB (LMANC LSB) – (0ffset 0x2F)

Bit	Name	Description
15:0	Reserved	Reserved.

3.4.5.1.6.2.18 LAN 0 MANC Value MSB (LMANC MSB) – (0ffset 0x30)

Bit	Name	Description
15:9	Reserved	Reserved.
8	Enable IPv4 Address Filters	When set, the last 128 bits of the MIPAF register are used to store four IPv4 addresses for IPv4 filtering. When cleared, these bits store a single IPv6 filter.
7	Enable XSUM Filtering to Manageability	When this bit is set, only packets that pass the L3 and L4 checksum are sent to the manageability block.
6	Reserved	Reserved.
5	Enable Manageability packets to Host Memory	This bit enables the functionality of the MANC2H register. When set, the packets that are specified in the MANC2H registers are also forwarded to host memory if they pass the manageability filters.
4:0	Reserved	Reserved.

3.4.5.1.6.2.19 LAN 0 Receive Enable 1 (LRXEN1) – (0ffset 0x31)

Bit	Name	Description
15:8	Receive Enable Byte 12	BMC SMBus slave address.
7	Enable BMC Dedicated MAC	
6	Reserved	Reserved. Must be set to 1b.
5:4	Notification Method	00b = SMBus alert. 01b = Asynchronous notify. 10b = Direct receive. 11b = Reserved.
3	Enable ARP Response	
2	Enable Status Reporting	
1	Enable Receive All	
0	Enable Receive TCO	

3.4.5.1.6.2.20 LAN 0 Receive Enable 2 (LRXEN2) – (0ffset 0x32)

Bit	Name	Description
15:8	Receive Enable byte 14	Alert Value
7:0	Receive Enable byte 13	Interface Data

3.4.5.1.6.2.21 LAN 0 MANC2H Value (LMANC2H LSB) – (0ffset 0x33)

Bit	Name	Description
15:8	Reserved	
7:0	Host Enable	When set, indicates that packets routed by the manageability filters to the manageability block are also sent to the host. Bit 0 corresponds to decision rule 0, etc.

3.4.5.1.6.2.22 LAN 0 MANC2H Value (LMANC2H MSB) – (0ffset 0x34)

Bit	Name	Description
15:0	Reserved	Reserved

3.4.5.1.6.2.23 Manageability Decision Filters- MDEF0 (1) – (0ffset 0x35)

Bit	Name	Description
15:12	Flex Port	Controls the inclusion of flexible port filtering in the manageability filter decision (OR section). Bit 12 corresponds to flexible port 0, etc. (see also bits 11:0 of the next word).
11	Port 0x26F	Controls the inclusion of port 0x26F filtering in the manageability filter decision (OR section).
10	Port 0x298	Controls the inclusion of port 0x298 filtering in the manageability filter decision (OR section).
9	Neighbor Discovery	Controls the inclusion of neighbor discovery filtering in the manageability filter decision (OR section).
8	ARP Response	Controls the inclusion of ARP response filtering in the manageability filter decision (OR section).
7	ARP Request	Controls the inclusion of ARP request filtering in the manageability filter decision (OR section).
6	Multicast	Controls the inclusion of multicast addresses filtering in the manageability filter decision (AND section).
5	Broadcast	Controls the inclusion of broadcast address filtering in the manageability filter decision (OR section).
4	Unicast	Controls the inclusion of unicast address filtering in the manageability filter decision (OR section).

Bit	Name	Description
3	IP Address	Controls the inclusion of IP address filtering in the manageability filter decision (AND section).
2	VLAN	Controls the inclusion of VLAN addresses filtering in the manageability filter decision (AND section).
1	Broadcast	Controls the inclusion of broadcast address filtering in the manageability filter decision (AND section).
0	Unicast	Controls the inclusion of unicast address filtering in the manageability filter decision (AND section).

3.4.5.1.6.2.24 Manageability Decision Filters- MDEF0 (2) – (0ffset 0x36)

Bit	Name	Description
15:12	Flexible TCO	Controls the inclusion of flexible TCO filtering in the manageability filter decision (OR section). Bit 12 corresponds to flexible TCO filter 0, etc.
11	Port 0x26F	Controls the inclusion of port 0x26F filtering in the manageability filter decision (OR section).
10:0	Flex port	Controls the inclusion of flexible port filtering in the manageability filter decision (OR section). Bit 11 corresponds to flexible port 0, etc. (see bits 15:12 of the previous word)

3.4.5.1.6.2.25 Manageability Decision Filters- MDEF1-6 (1-2) – (0ffset 0x37-0x42)

Same as words 0x035 and 0x36 for MDEF1-MDEF6.

3.4.5.1.6.2.26 ARP Response IPv4 Address 0 (LSB) – (0ffset 0x43)

Bit	Name	Description
15:8	ARP Response IPv4 Address 0, Byte 1	
7:0	ARP Response IPv4 Address 0, Byte 0	

3.4.5.1.6.2.27 ARP Response IPv4 Address 0 (MSB) – (0ffset 0x44)

Bit	Name	Description
15:8	ARP Response IPv4 Address 0, Byte 3	
7:0	ARP Response IPv4 Address 0, Byte 2	

3.4.5.1.6.2.28 LAN 0 IPv6 Address 0 (LSB) (MIPAF) – (0ffset 0x45)

Bit	Name	Description
15:8	LAN 0 IPv6 Address 0, Byte 1	
7:0	LAN 0 IPv6 Address 0, Byte 0	

3.4.5.1.6.2.29 LAN 0 IPv6 Address 0 (MSB) (MIPAF) – (0ffset 0x46)

Bit	Name	Description
15:8	LAN 0 IPv6 Address 0, Byte 3	
7:0	LAN 0 IPv6 Address 0, Byte 2	

3.4.5.1.6.2.30 LAN 0 IPv6 Address 0 (LSB) (MIPAF) – (0ffset 0x47)

Bit	Name	Description
15:8	LAN 0 IPv6 Address 0, Byte 5	
7:0	LAN 0 IPv6 Address 0, Byte 4	

3.4.5.1.6.2.31 LAN 0 IPv6 Address 0 (MSB) (MIPAF) – (0ffset 0x48)

Bit	Name	Description
15:8	LAN 0 IPv6 Address 0, Byte 7	
7:0	LAN 0 IPv6 Address 0, Byte 6	

3.4.5.1.6.2.32 LAN 0 IPv6 Address 0 (LSB) (MIPAF) – (0ffset 0x49)

Bit	Name	Description
15:8	LAN 0 IPv6 Address 0, Byte 9	
7:0	LAN 0 IPv6 Address 0, Byte 8	

3.4.5.1.6.2.33 LAN 0 IPv6 Address 0 (MSB) (MIPAF) – (0ffset 0x4A)

Bit	Name	Description
15:8	LAN 0 IPv6 Address 0, Byte 11	
7:0	LAN 0 IPv6 Address 0, Byte 10	

3.4.5.1.6.2.34 LAN 0 IPv6 Address 0 (LSB) (MIPAF) – (0ffset 0x4B)

Bit	Name	Description
15:8	LAN 0 IPv6 Address 0, Byte 13	
7:0	LAN 0 IPv6 Address 0, Byte 12	

3.4.5.1.6.2.35 LAN 0 IPv6 Address 0 (MSB) (MIPAF) – (0ffset 0x4C)

Bit	Name	Description
15:8	LAN 0 IPv6 Address 0, Byte 15	
7:0	LAN 0 IPv6 Address 0, Byte 14	

3.4.5.1.6.2.36 LAN 0 IPv6 Address 1 (MIPAF) – (0ffset 0x4D-0x54)

Same structure as LAN 0 IPv6 Address 0.

3.4.5.1.6.2.37 LAN 0 IPv6 Address 2 (MIPAF) – (0ffset 0x55-0x5C)

Same structure as LAN 0 IPv6 Address 0.

3.4.5.1.7 SMBus Configuration Structure

3.4.5.1.7.1 Section Header – (0ffset 0x0)

Bit	Name	Description
15:8	Block CRC8	
7:0	Block Length	

3.4.5.1.7.2 SMBus Maximum Fragment Size – (0ffset 0x01)

Bit	Name	Description
15:0	SMBus Maximum Fragment Size (bytes)	

3.4.5.1.7.3 SMBus Notification Timeout and Flags – (0ffset 0x02)

Bit	Name	Description
15:8	SMBus Notification Timeout (ms)	
7:6	SMBus Connection Speed	00b = Slow SMBus connection. 01b = Reserved. 10b = Reserved. 11b = Reserved.
5	SMBus Block Read Command	0b = Block read command is 0xC0. 1b = Block read command is 0xD0.
4	SMBus Addressing Mode	0b = Single address mode. 1b = Dual address mode.
3	Reserved	Reserved.
2	Disable SMBus ARP Functionality	
1	SMBus ARP PEC	
0	Reserved	Reserved.

3.4.5.1.7.4 SMBus Slave Addresses – (0ffset 0x03)

Bit	Name	Description
15:9	SMBus 1 Slave Address	Dual address mode only.
8	Reserved	Reserved.
7:1	SMBus 0 Slave Address	
0	Reserved	Reserved.

3.4.5.1.7.5 Fail-Over Register (Low Word) – (0ffset 0x04)

Bit	Name	Description
15:12	Gratuitous ARP Counter	
11:10	Reserved	Reserved.
9	Enable Teaming Fail-Over on DX	
8	Remove Promiscuous on DX	
7	Enable MAC Filtering	
6	Enable Repeated Gratuitous ARP	
5	Reserved	Reserved.
4	Enable Preferred Primary	
3	Preferred Primary Port	
2	Transmit Port	
1:0	Reserved	Reserved.

3.4.5.1.7.6 Fail-Over Register (High Word) – (0ffset 0x05)

Bit	Name	Description
15:8	Gratuitous ARP Transmission Interval (seconds)	
7:0	Link Down Fail-Over Time	

3.4.5.1.7.7 NC-SI Configuration (0ffset 0x06)

Bit	Name	Description
15:8	Reserved	Reserved.
7:5	Package ID	
4:0	LAN0 Internal Channel ID	

3.4.5.1.8 Flexible TCO Filter Configuration Structure

3.4.5.1.8.1 Section Header – (0ffset 0x0)

Bit	Name	Description
15:8	Block CRC8	
7:0	Block Length	

3.4.5.1.8.2 Flexible Filter Length and Control – (0ffset 0x01)

Bit	Name	Description
15:8	Flexible Filter Length (bytes)	
7:5	Reserved	Reserved.
4	Last Filter	
3:2	Filter Index (0-3)	
1	Apply Filter to LAN 1	
0	Apply Filter to LAN 0	

3.4.5.1.8.3 Flexible Filter Enable Mask – (0ffset 0x02 – 0x09)

Bit	Name	Description
15:0	Flexible Filter Enable Mask	

3.4.5.1.8.4 Flexible Filter Data – (0ffset 0x0A – Block Length)

Bit	Name	Description
15:0	Flexible Filter Data	

3.4.5.1.9 NC-SI Microcode Download Structure

3.4.5.1.9.1 Patch Data Size – (Offset 0x0)

Bit	Name	Description
15:0	Data Size	

3.4.5.1.9.2 Rx and Tx Code Size – (Offset 0x1)

Bit	Name	Description
15:8	Rx Code Length in Dwords	
7:0	Tx Code Length in Dwords	

3.4.5.1.9.3 Download Data – (Offset 0x2 – Data Size)

Bit	Name	Description
15:0	Download Data	

3.4.5.1.10 NC-SI Configuration Structure

3.4.5.1.10.1 Section Header (0ffset 0x0)

Bit	Name	Description
15:8	Block CRC8	
7:0	Block Length	

3.4.5.1.10.2 Rx Mode Control1 (RR_CTRL[15:0]) (Offset 0x1)

Bit	Name	Description
15:8	Reserved	Reserved. Should be set to 0b.
7:4	Reserved	Reserved.
3	NC-SI Speed	When set, the NC-SI MAC speed is 100 Mb/s. When reset, NC-SI MAC speed is 10 Mb/s.
2	Receive Without Leading Zeros	If set, packets without leading zeros (J/K/ symbols) between TXEN assertion and TXD the first preamble byte can be received.
1	Clear Rx Error	Should be set when the Rx path is stuck because of an overflow condition.
0	NC-SI Loopback Enable	When set, enables NC-SI TX to RX loop. All data that is transmitted from NC-SI is returned to it. No data is actually transmitted from NC-SI.

3.4.5.1.10.3 Rx Mode Control2 (RR_CTRL[31:16]) (Offset 0x2)

Bit	Name	Description
15:0	Reserved	Reserved. Should be set to 0b.

3.4.5.1.10.4 Tx Mode Control1 (RT_CTRL[15:0]) (Offset 0x3)

Bit	Name	Description
15:3	Reserved	Reserved. Should be set to 0b.
2	Transmit With Leading Zeros	When set, sends leading zeros (J/K/ symbols) from CRS_DV assertion to the start of preamble (PHY Mode). When deasserted, does not send leading zeros (MAC mode).
1	Clear Tx Error	Should be set when Tx path is stuck because of an underflow condition. Cleared by hardware when released.
0	Enable Tx Pads	When set, the NC-SI TX pads are driving; otherwise, they are isolated.

3.4.5.1.10.5 Tx Mode Control2 (RT_CTRL[31:16]) (Offset 0x4)

Bit	Name	Description
15:0	Reserved	Reserved. Should be 0b.

3.4.5.1.10.6 MAC Tx Control Reg1 (TxCntrlReg1 (15:0]) (Offset 0x5)

Bit	Name	Description
15:7	Reserved	Reserved. Should be set to 0b.
6	NC-SI_enable	Enable the MAC internal NC-SI mode of operation (disables external NC-SI gasket).
5	Two_part_deferral	When set, performs the optional two part deferral.
4	Append_fcs	When set, computes and appends the FCS on Tx frames.
3	Pad_enable	Pad the TX frames, which are less than the minimum frame size.
2:1	Reserved	Reserved.
0	Tx_ch_en	Tx Channel Enable This bit can be used to enable the Tx path of the MAC. This bit is for debug only and the recommended way to enable the Tx path is via the RT_UCTL_CTRL .TX_enable bit.

3.4.5.1.10.7 NC-SI Settings (NCSISET) (Offset 0x7)

Bit	Name	Description
15:2	Reserved	Reserved. Should be set to 0b.
1	NC-SI Out Buffer Strength	0b = Value 0 should be configured to the NC-SI out buffer strength. 1b = Value 1 should be configured to the NC-SI out buffer strength
0	NC-SI In Buffer Strength	0b = Value 0 should be configured to the NC-SI in buffer strength. 1b = Value 1 should be configured to the NC-SI in buffer strength.

3.5 Rx/Tx Functions

3.5.1 Device Data/Control Flows

This section describes both the transmit and receive data flows for the 82598.

3.5.1.1 Transmit Data Flow

Tx data flow provides a high-level description of all data/control transformation steps needed for sending Ethernet packets over the wire.

Step	Description
1	The host creates a descriptor ring and configures one of the 82598's transmit queues with the address location, length, head, and tail pointers of the ring (one of 32 available Tx queues).

2	The host transmits a packet provided by the TCP/IP stack. This packet arrives in one or more data buffers.
3	The host initializes the descriptor(s) that point to the data buffer(s) and adds additional control parameters that describe the needed hardware functionality. The host places that descriptor in the correct location in the appropriate Tx ring.
4	The host updates the appropriate Queue Tail Pointer (TDT).
5	The 82598'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.
7	The DMA fetches the next descriptor and processes its contents. As a result, the DMA sends PCIe requests to fetch the packet data from system memory.
8	The packet data is being received from PCIe read completions and passes through the transmit DMA, which performs all programmed data manipulations on the packet data on the fly. These can include various CPU offloading tasks such as TSO offloading and checksum offloads.
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 then 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 over the wire using a pre-configured interface.
12	When all the PCIe completions for a given packet are complete, the DMA updates the appropriate descriptor(s).
13	The descriptors are written back to host memory using PCIe posted writes.
14	An interrupt is generated to notify the host driver that the specific packet has been read to the 82598 and the driver can then release the buffer(s).

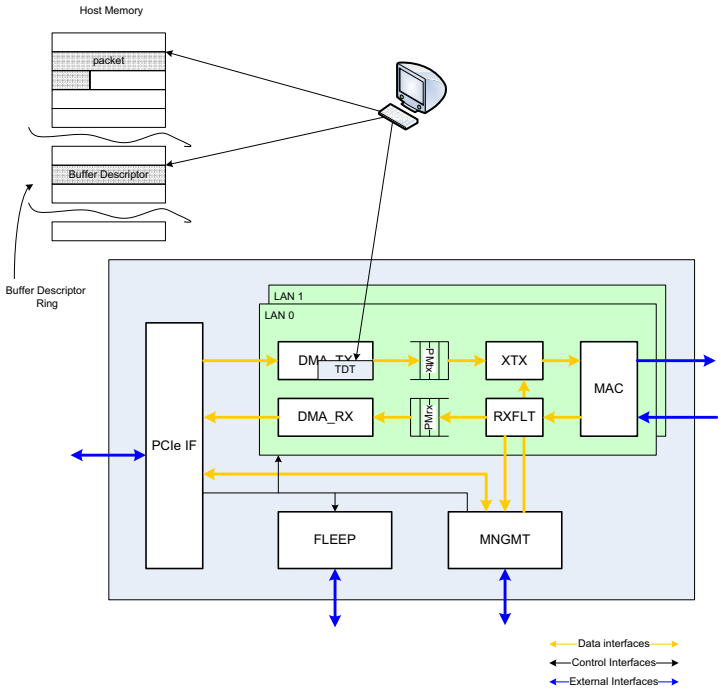


Figure 3-18. Transmit Data Flow

3.5.1.2 Rx Data Flow

Rx data flow provides a high-level description of all data/control transformation steps needed for receiving Ethernet packets.

Step	Description
1	The host creates a descriptor ring and configures one of the 82598's receive queues with the address location, length, head, and tail pointers of the ring (one of 64 available Rx queues)
2	The host initializes descriptor(s) that point to empty data buffer(s). The host places these descriptor(s) in the correct location at the appropriate Rx ring.
3	The host updates the appropriate Queue Tail Pointer (RDT).
4	The 82598's DMA senses a change of a specific RDT and as a result sends a PCIe request to fetch the descriptor(s) from host memory.
5	The descriptor(s) content is received in a PCIe read completion and is written to the appropriate location in the descriptor queue.
6	A packet enters the Rx MAC.
7	The MAC forwards the packet to the Rx filter.
8	If the packet matches the pre-programmed criteria of the Rx filtering, it is forwarded to the Rx FIFO.
9	The receive DMA fetches the next descriptor from the appropriate ring to be used for the next received packet.

Step	Description
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 the packet data.
12	The receive DMA writes back the descriptor content along with status bits that indicate the packet information including what offloads were done on that packet.
13	The 82598 initiates an interrupt to the host to indicate that a new received packet is ready in host memory.
14	The host reads the packet data and sends it to the TCP/IP stack for further processing. The host releases the associated buffer(s) and descriptor(s) once they are no longer in use.

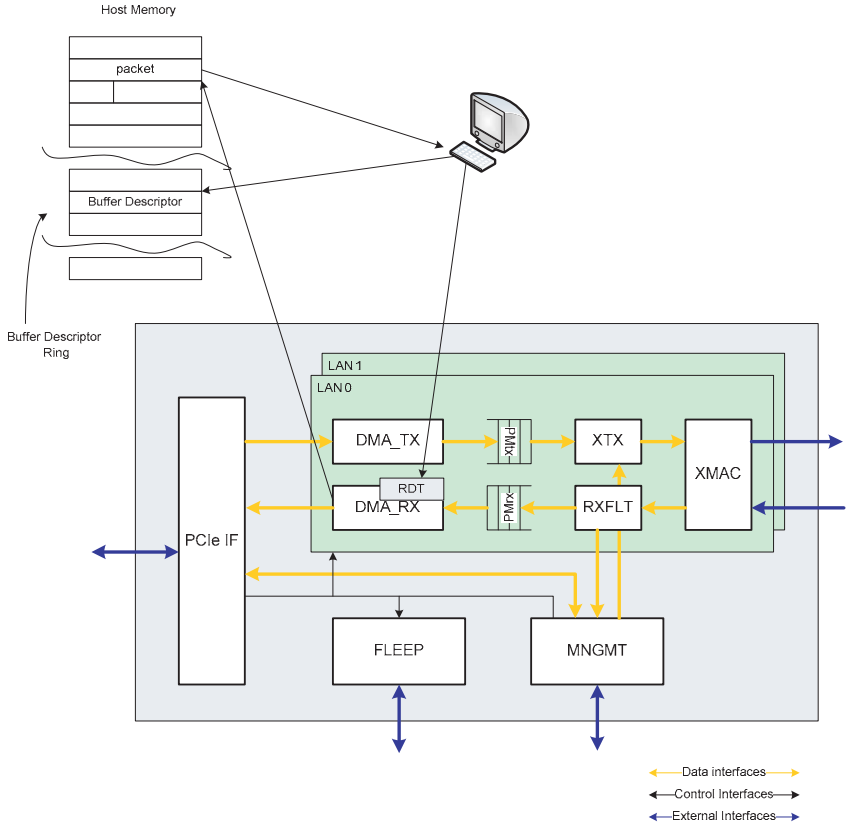
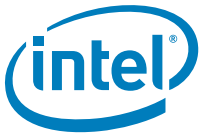


Figure 3-19. Receive Data Flow

3.5.2 Receive Functionality

Packet reception consists of recognizing the presence of a packet on the wire, performing address filtering and DMA queue assignment, storing the packet in the receive data FIFO, transferring the data to assigned receive queues in host memory, and updating the state of a receive descriptor.



As a receive packet is accepted and processed, it is stored in on-die packet buffers before being transferred to system memory. The 82598 supports up to eight separate packet buffers.

Each of the active packet buffers has one or more receive descriptor queues assigned to it. The descriptor queues operate independently (but under a central arbitration scheme) to forward receive packets from their packet buffers into system memory. The following section describes how the 82598's receive descriptor queues are assigned.

The following operational modes impact allocation of receive descriptor queues:

- RSS (Receive Side Scaling) – RSS shares packet processing between several processor cores by assigning packets into different descriptor queues. RSS assigns each packet an RSS output index. See Section 3.5.2.10 for details on RSS.
- Virtual Machine Device queues (VMDq) – VMDq shares the 82598 DMA resources between more than one software entity (operating system and/or software device driver). This is done through replication of receive descriptor queues and their configuration registers. Current uses of VMDq are for virtualized environments. VMDq assigns each packet a VMDq output index. See Section 3.5.2.10.3 for more details on VMDq.

Table 3-47. Supported Modes for Allocation of Receive Descriptor Queues

Single Packet Buffer			
		RSS	
		Off	On
VMDq	Off	Supported	Supported
	On	Supported	Supported

Since the 82598 provides a total of 64 receive descriptor queues, a received packet is assigned a 6-bit queue index. Each of RSS and VMDq determines the value of some bits in the queue index:

- RSS provides a 4-bit RSS output index. Queue allocation can use of four bits or just some LSB of it. RSS assigns an RSS output index equal to zero for traffic that does not go through RSS. Depending on the configuration of VMDq/RSS, such traffic might end up in one of several queues.
- VMDq provides a 4-bit VMDq output index. Queue allocation can make use of four bits or just some LSB of it.

Generating a 6-bit queue index from the RSS and VMDq indices is defined in Table 3-48 and the notes that follow it.

Table 3-48. Queue Assignment for Received Packets

	Configuration		Bit Allocation [5:0]		
Case	VMDq	RSS	VMDq	RSS	Comments
1	Off	Off	None	None	Hardware default. Queue 0 used
2	Off	On	None	3:0	Bits [5:4]=00b
3	On	Off	3:0	None	Bits [5:4]=00b
4	On	On	5:4	3:0	

Note:

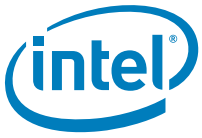
- Case 1 – A single receive packet buffer with a single queue 0 is used for all receive traffic.
- Case 2 – A single receive packet buffer is used. RSS determines one of up to 16 queues per receive packet
- Case 3 – A single receive packet buffer is used. VMDq determines one of up to 16 queues per receive packet
- Case 4 – Used to separate traffic into four sets, each with 16 queues. A queue is selected as follows:
  - Bits 5:4 of the queue index are provided from bits 1:0 of the VMDq output index.
  - Bits 3:0 are provided from the RSS output index. If bit 0 of the VMDq output index is 0b, then RSS output index 0 is used. If bit 0 of the VMDq output index is 1b, then RSS output index 1 is used.

Configuration registers (CSRs) that control queue operation are replicated per queue (total of 64 copies of each register). Each of the replicated registers corresponds to a queue such that the 6-bit queue index equals the serial number of the register (register 0 corresponds to queue 0, etc.). Registers included in this category are: RDBAL[63:0] and RDBAH[63:0] – Rx Descriptor Base

- RDLEN[63:0] – RX Descriptor Length
- RDH[63:0] – RX Descriptor Head
- RDT[63:0] – RX Descriptor Tail
- RXDCTL[63:0] – Receive Descriptor Control

Configuration registers (CSRs) that define the functionality of descriptor queues are replicated per VMDq index to allow for separate configuration in a virtualization environment (total of 16 copies of each register). Each of the replicated registers corresponds to a set of queues with the same VMDq index, such that the VMDq index of the queue identifies the serial number of the register. Examples:

- Case 3 above – The VMDq index defines bits [3:0] of the queue index (all 16 copies are used, one per value of the VMDq index). Therefore, queue 0 is associated with the register indexed 0, queue 1 is associated with the register indexed 1, etc.
- Case 4 above – The VMDq index defines bits [5:4] of the queue index (only 4 copies are used, one per value of the VMDq index). Therefore, queues 0, 1, …, 15 are associated with the register indexed 0, queues 16, 17, …, 31 are associated with the register indexed 1, etc.
- Else – a single copy of the following registers is used (register # 0).



Registers included in this category are:

- DCA_RXCTRL[15:0] – Rx DCA Control
- SRRCTL[15:0] – Split and Replication Receive Control
- PSRTYPE[15:0] – Packet Split Receive Type

3.5.2.1 Packet Filtering

The receive packet filtering role is determining which of the incoming packets are allowed to pass to the local system, and which should be dropped. Received packets can be destined to the host, to a BMC, or to both. This section describes how host filtering is done, and the interaction with management filtering.

Note: Maximum supported received packet size is 16 kB.

As shown in Figure 3-20, host filtering is done in three stages:

1. Packets are filtered by L2 filters (MAC address, unicast/multicast/broadcast). See Section 3.5.2.1.1 for details.
2. Packets are then filtered by VLAN filters if a VLAN tag is present. See Section 3.5.2.1.2 for details.
3. Packets are filtered by the manageability filters (port, IP, flex, other). Refer to the *Intel® 82598 10 GbE Controller System Manageability Interface* application note for details.

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

1. The packet does not pass L2 filters.

- The packet does not pass VLAN filtering.
- The packet passes manageability filtering and the manageability filters determine that the packet should not pass to host as well. Refer to the *Intel® 82598 10 GbE Controller System Manageability Interface* application note for details.

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, Packet Error, Length Error and CRC Error detected. However, if the *Store-Bad-Packet* (SBP) bit is set (FCTRL.SBP), then bad packets that don't 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’re bad, by setting promiscuous enables and the *SBP* bit.

Note: CRC errors before the SFD are ignored. Any packet must have a valid SFD in order to be recognized by the 82598 (even bad packets).

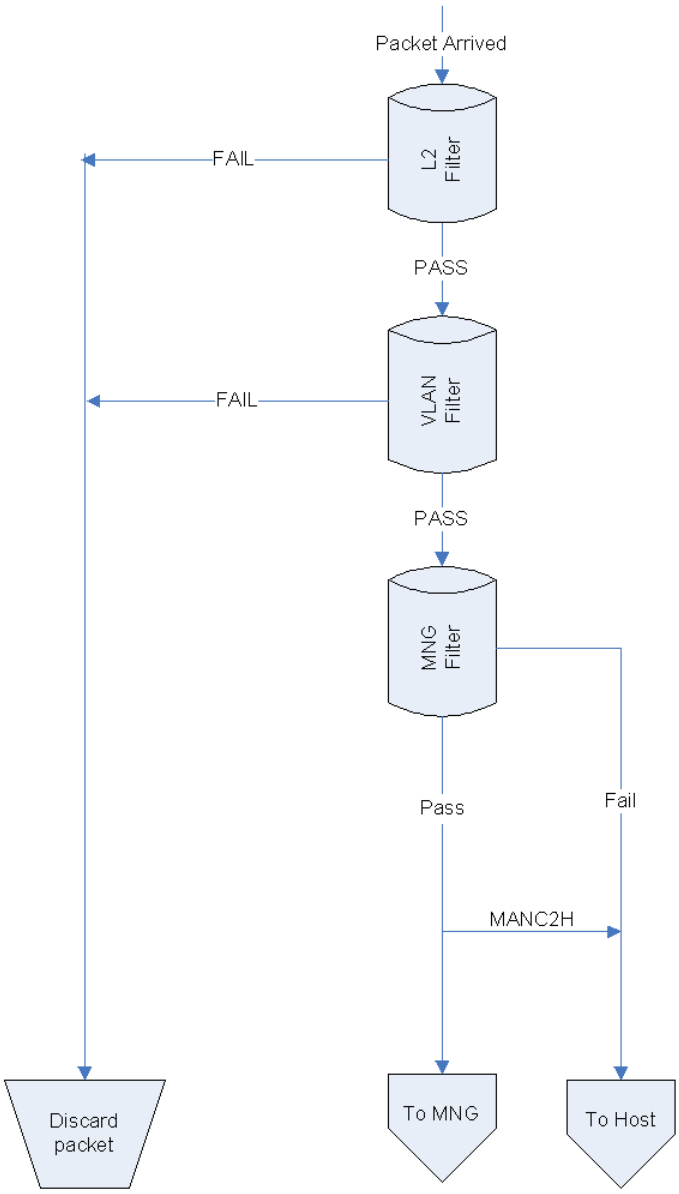


Figure 3-20. Rx Filtering Flow Chart

3.5.2.1.1 L2 Filtering

Figure 3-21 shows L2 filtering. A packet passes successfully through L2 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 (host or manageability).

4. It is a multicast packet and it matches one of the multicast filters.
5. It is a broadcast packet and Broadcast Accept Mode (BAM) is enabled. Note that in this case, for manageability traffic the packet does not go through VLAN filtering (VLAN filtering is assumed to match).

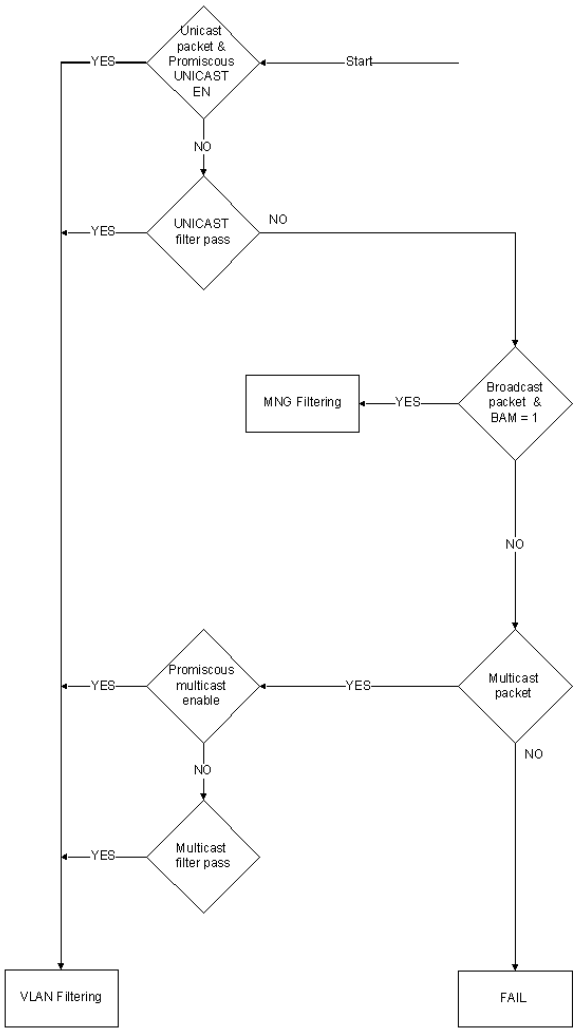


Figure 3-21. L2 Rx Filtering Flow Chart

3.5.2.1.1.1 Unicast Filter

The entire MAC address is checked against the 16 host unicast addresses and four management unicast addresses (if enabled). The 16 host unicast addresses are controlled by the host interface (the BMC must not change them). The other four addresses are dedicated to management functions and are only accessed by the BMC. The destination address of incoming packets must exactly match one of the pre-configured host address filters or the manageability address filters. These addresses can be unicast or multicast. Those filters are configured through Receive Address Low – RAL (0x05400 + 8*n[n=0..15]; RW), Receive Address High – RAH (0x05404 + 8*n[n=0..15]; RW), Manageability MAC Address Low – MMAL (0x5910 + 8*n[n=0..3]; RW) and Manageability MAC Address High – MMAH (0x5914 + 8*n[n=0..3]; RW) registers.

Promiscuous Unicast – Receive all unicasts. Promiscuous unicast mode can be set/cleared only through the host interface (not by the BMC), and it is usually used when the 82598 is used as a sniffer.

3.5.2.1.1.2 Multicast Filter (Partial)

The 12-bit portion of an incoming packet multicast address must exactly match Multicast Filter Address in order to pass the Multicast Filter. Those 12 bits out of the 48 bits of Destination Address can be selected by the *MO* field. These entries can be configured only by the host interface and cannot be controlled by the BMC.

Promiscuous Multicast – Receive all multicasts. Promiscuous multicast mode can be set/cleared only through the host interface (not by the BMC), and it is usually used when the 82598 is used as a sniffer.

3.5.2.1.2 VLAN Filtering

Figure 3-22 shows VLAN filtering. A receive packet that successfully passed L2 layer 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.
2. If the packet has a VLAN header and it passes a valid manageability VLAN filter, then is passes to the next filtering stage.
3. If the packet has a VLAN header, it did not match step 2, and no host VLAN filters are enabled, the packet is forwarded to the next filtering stage.
4. If the packet has a VLAN header, it did not match step 2, and it matches an enabled host VLAN filter, the packet is forwarded to the next filtering stage.
5. Otherwise, the packet is dropped.

The 82598 provides exact VLAN filtering for VLAN tags for host traffic and VLAN tags for manageability traffic. See VLAN Filter Table Array – VFTA (0x0A000-0x0A9FC; RW) for detailed information on programming VLAN filters.

The BMC configures the 82598 with eight different manageability VIDs via the Management VLAN TAG Value [7:0] – MAVTV[7:0] registers and enables each filter in the MFVAL register.

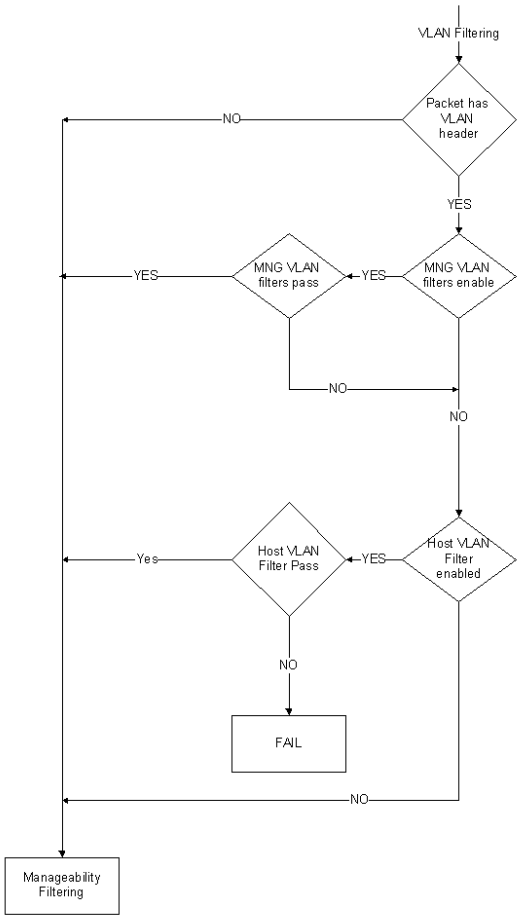


Figure 3-22. VLAN Filtering

3.5.2.2 *Intel® 82598 10 GbE Controller System Manageability Interface* application note**Receive Data Storage**

The descriptor points to a memory buffer to store packet data.

The size of the buffer is set using the per queue SRRCTL[n].BSIZEPACKET field.

Receive buffer size selected by bit settings SRRCTL[n].BSIZEPACKET support buffer sizes of 1 kB granularity.

In addition, for advanced descriptor usage the SRRCTL.BSIZEHEADER field is used to define the size of the buffers allocated to headers.

The 82598 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.

Although alignment is completely unrestricted, it is highly recommended that software allocate receive buffers on at least cache-line boundaries whenever possible.

Note: When the *No-Snoop Enable* bit is used in advanced descriptors, the buffer address should always be 16-bit aligned.

3.5.2.3 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 82598 uses the Legacy Rx Descriptor as listed in Table 3-49. The shaded areas indicate fields that are modified by hardware upon packet reception (called descriptor write-back).

Table 3-49. 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 0	Fragment Checksum[1]	Length

1. The checksum indicated here is the unadjusted 16-bit ones complement of the packet. A software assist might be required to back out appropriate information prior to sending it up to upper software layers. The fragment checksum is always reported in the first descriptor (even in the case of multi-descriptor packets).

Length Field (16 bit offset 0)

After receiving a packet for the 82598, hardware stores the packet data into the indicated buffer and writes the length, status, errors, and status fields. 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.

Fragment Checksum (16 bit offset 16)

This field is used to provide the fragment checksum value.

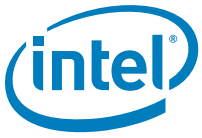
Status 0 Field (8 bit offset 32)

Status information indicates whether the descriptor has been used and whether the referenced buffer is the last one for the packet. Refer to Table 3-50 for the layout of the status field. Error status information is shown in Table 3-51.

Table 3-50. Receive Status 0 (RDESC.STATUS-0) Layout

7	6	5	4	3	2	1	0
PIF	IPCS	L4CS	UDPCS	VP	Reserved	EOP	DD

- PIF (bit 7) – Passed in-exact filter
- IPCS (bit 6) – IPv4 checksum calculated on packet
- L4CS (bit 5) – L4 checksum calculated on packet



- UDPCS (bit 4) – UDP/TCP checksum calculated on packet
- VP (bit 3) – Packet is 802.1q (matched VET); indicates strip VLAN in 802.1q packet
- Reserved (bit 2) – Reserved
- EOP (bit 1) – End of packet
- DD (bit 0) – Descriptor done

EOP, DD

Packets that exceed the receive buffer size span multiple receive buffers. EOP indicates whether this is the last buffer for an incoming packet. DD indicates whether hardware is done with the descriptor. When set along with EOP, the received packet is complete in main memory. Software can determine buffer usage by setting the status byte to zero before making the descriptor available to hardware, and checking it for non-zero content at a later time. For multi-descriptor packets, packet status is provided in the final descriptor of the packet (EOP set). If EOP is not set for a descriptor, only the Address, Length, and DD bits are valid.

VP

The VP field indicates whether the incoming packet's type matches VET (if the packet is a VLAN (802.1q) type). It's set if the packet type matches VET and VLNCTRL.VME is set. It also indicates that VLAN has been stripped in the 802.1q packet. For a further description of 802.1q VLANs please see Section 3.5.5.

IPCS, L4CS, UDPCS: The meaning of these bits is shown in the following table:

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 active and TCP checksum offload. Pass/Fail indication is provided in the Error field – IPE and TCPE. See PKTTYPE table for supported packet types.
1b	1b	1b/0b	Hardware provides IPv4 checksum offload if IPCS is active along with UDP checksum offload. Pass/fail indication is provided in the Error field – IPE and TCPE. See PKTTYPE table for supported packet types.

See Section 3.5.2.12 for a description of supported packet types for receive checksum offloading. IPv6 packets do not have the IPCS bit set, but might have the L4CS bit set if the 82598 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 set to determine whether to accept the packet. If PIF is clear, then the packet is known to be for this station, so software need not look at the packet contents. Packets passing only the Multicast Vector (MTA) but not any of the MAC address exact filters (RAH, RAL) has PIF set.

Error Field (8 bit offset 40)

Most error information appears only when the *SBP* bit (FCTRL.SBP) is set and a bad packet is received. Refer to Table 3-51 for a definition of the possible errors and their bit positions.

Table 3-51. Receive Errors (RDESC.ERRORS) Layout

7	6	5	4	3	2	1	0
IPE	TCPE	USE	OSE	PE	Reserved	LE	CE

- IPE (bit 7) – IPv4 checksum error
- TCPE (bit 6) – TCP/UDP checksum error
- USE (bit 5) – Undersize error – Undersized packet received
- OSE (bit 4) – Oversize error – Oversized packet received
- PE (bit 3) – Packet error – Illegal symbol or error symbol in the middle of a received packet
- Reserved (bit 2) – Reserved
- LE (bit 1) – Length error
- CE (bit 0) – CRC error

The IP and TCP checksum error bits from Table 3-51 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.

VLAN Tag Field (16 bit offset 48)

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 VLNCTRL.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.

Table 3-52. VLAN Tag Field Layout (for 802.1q Packet)

15 13	12	11 0
PRI	CFI	VLAN

3.5.2.4 Advanced Receive Descriptors

Table 3-53 lists the receive descriptor.

Table 3-53. 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)

The physical address of the packet buffer. The lowest bit is either A0 (LSB of address) or NSE (No Snoop Enable), depending on bit DCA_RXCTRL.RXdataWriteNSEn of the relevant queue.

Header Buffer Address (64)

The physical address of the header buffer. the lowest bit is DD (Descriptor done).

Note: The 82598 does not support null descriptors, in which a packet or header address is equal to zero.

When software sets the NSE (No-snoop) bit, the 82598 places the received packet associated with this descriptor in memory at the packet buffer address with the no-snoop bit 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.

No-snoop 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 data movement 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 operation with the packet.

Note: When No-Snoop Enable is used, relaxed ordering should also be enabled with CTRL_EXT.RO_DIS.

Note that when the 82598 writes back the descriptors, it uses the descriptor format shown in Table 3-54. The SRRCTL[n].DESCTYPE must be set to a value other than 000b for the 82598 to write back the special descriptors.

Table 3-54. Descriptor Write-Back Format

<table>
<tr><td></td><td>63
60</td><td>59
56</td><td>55
52</td><td>51
48</td><td>47
44</td><td>43
40</td><td>39
36</td><td>35
32</td><td>31</td><td>30:21</td><td>20
16</td><td>15:4</td><td>3:0</td></tr>
<tr><td>0</td><td colspan="8">RSS Hash value</td><td rowspan="2">SPH</td><td rowspan="2">HDR_buf_len</td><td rowspan="2">RSV</td><td rowspan="2">Packet Type</td><td rowspan="2">RSS Type</td></tr>
<tr><td></td><td colspan="4">Fragment Checksum</td><td colspan="4">IP Identification</td></tr>
<tr><td>8</td><td colspan="4">VLAN Tag</td><td colspan="4">PKT_buf_Length</td><td colspan="2">Extended Error</td><td colspan="3">Extended Status</td></tr>
<tr><td></td><td colspan="4">63</td><td colspan="4">48</td><td colspan="4">47</td><td colspan="4">32</td><td>31</td><td>20</td><td>19</td><td colspan="2">0</td></tr>
</table>

Light blue fields are mutually exclusive by RXCSUM.PCSD.

Packet Type (12)

- RSV(11-8) – Reserved
- NFS (bit 7) – NFS header present
- SCTP (bit 6) – SCTP header present
- UDP (bit 5) – UDP header present
- TCP (bit 4) – TCP header present

- IPV6E (bit 3) – IPv6 header includes extensions
- IPv6 (bit 2) – IPv6 header present
- IPv4E (bit1) – IPv4 header includes extensions
- IPv4 (bit 0) – IPv4 header present

Note: UDP, TCP and IPv6 indication are not set in an IPv4 fragmented packet.

RSS Type (8)

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
0xA7 – 0xF	Reserved

The 82598 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.

RSV(5)

Reserved.

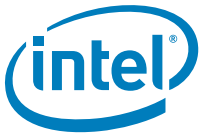
Split Header (11)

- SPH(bit 9) – When set to 1b, indicates that the hardware has found the length of the header. If set to 0b, the *Header Buffer Length* field is ignored.
- HDR_BUF_LEN(bit 9:0) – The length (bytes) of the header as parsed by the 82598.

In header split mode (SPH set to 1b), this field also reflects the size of the header that was actually stored in the buffer. However, in header replication mode (SPH is also set in this mode), this does not reflect the size of the data actually stored in the header buffer. This is because the 82598 fills the buffer up to the size configured by SRRCTL[n].BSIZEHEADER that might be larger than the header size reported here.

Note: Packets that have headers larger than 1 kB are not split.

Packet types supported by the packet split: the 82598 provides header split for the packet types that follow. Other packet types are posted sequentially in the host packet buffer.



Each line in the following table has an enable bit in the PSRTYPE register. When one of the bits is set, the corresponding packet type is split.

Packet Type	Description	Header Split
0x0	Header includes MAC, (VLAN/SNAP) only.	No.
0x1	Header includes MAC, (VLAN/SNAP), IPv4, Only	Split header after L3 if fragmented packets.
0x2	Header includes MAC, (VLAN/SNAP), IPv4, TCP, only	Split header after L4 if not fragmented, otherwise treat as packet type 1.
0x3	Header includes MAC, (VLAN/SNAP), IPv4, UDP only	Split header after L4 if not fragmented, otherwise treat as packet type 1.
0x4	Header includes MAC (VLAN/SNAP), IPv4, IPv6, only	Split header after L3 if IPv6 indicates a fragmented packet or treat as packet type 0x1 if IPv4 header is fragmented.
0x5	Header includes MAC (VLAN/SNAP), IPv4, IPv6,TCP, only	Split header after L4 if IPv4 not fragmented and if IPv6 does not include fragment extension header, otherwise treat as packet type 4.
0x6	Header includes MAC (VLAN/SNAP), IPv4, IPv6, UDP only	Split header after L4 if IPv4 not fragmented and if IPv6 does not include fragment extension header, otherwise treat as packet type 4.
0x7	Header includes MAC (VLAN/SNAP), IPv6, only	Split header after L3 if fragmented packets.
0x8	Header includes MAC (VLAN/SNAP), IPv6, TCP, only	Split header after L4 if IPv6 does not include fragment extension header, otherwise treat as packet type 7.
0x9	Header includes MAC (VLAN/SNAP), IPv6, UDP only	Split header after L4 if IPv6 does not include fragment extension header, otherwise treat as packet type 7.
0xA	Reserved	
0xB	Header includes MAC, (VLAN/SNAP) IPv4, TCP, NFS, only	Split header after L5 if not fragmented, otherwise treat as packet type 1.
0xC	Header includes MAC, (VLAN/SNAP), IPv4, UDP, NFS, only	Split header after L5 if not fragmented, otherwise treat as packet type 1.
0xD	Reserved	
0xE	Header includes MAC (VLAN/SNAP), IPv4, IPv6, TCP,NFS, only	Split header after L5 if IPv4 not fragmented and if IPv6 does not include fragment extension header, otherwise treat as packet type 4.
0xF	Header includes MAC (VLAN/SNAP), IPv4, IPv6, UDP, NFS, only	Split header after L5 if IPv4 not fragmented and if IPv6 does not include fragment extension header, otherwise treat as packet type 4.
0x10	Reserved	
0x11	Header includes MAC (VLAN/SNAP), IPv6, TCP, NFS, only	Split header after L5 if IPv6 does not include fragment extension header, otherwise treat as packet type 7.
0x12	Header includes MAC (VLAN/SNAP), IPv6, UDP, NFS, only	Split header after L5 if IPv6 does not include fragment extension header, otherwise treat as packet type 7.

Note: Header of fragmented IPv6 packet is defined until the fragmented extension header.

Fragment Checksum (16)

This field is used to provide the fragment checksum value for fragmented UDP packets. The *Fragment Checksum* field can be used to accelerate UDP checksum verification by the host processor. This operation is enabled by the RXCSUM.IPPCSE bit.

This field is mutually exclusive with the RSS hash. It is enabled when the RXCSUM.PCSD bit is cleared.

RSS Hash Value (32)

RSS hash value.

Extended Status (20)

- Reserved (bits 19:12) – Reserved
- DYNINT (bit 11) – Packet caused immediate interrupt via dynamic interrupt moderation
- UDPV (bit 10) – Valid UDP checksum
- Reserved (bit 9) – Reserved
- CRCV (bit 8) – Speculative CRC valid
- PIF (bit 7) – Passed inexact filter
- IPCS (bit 6) – IPv4 checksum calculated on packet
- L4CS (bit 5) – TCP checksum calculated on packet
- UDPCS (bit 4) – UDP checksum calculated on packet
- VP (bit 3) – Packet is 802.1q (matched VET). Indicates strip VLAN field.
- Reserved (bit 2) – Reserved
- EOP (bit 1) – End of packet
- DD (bit 0) – Descriptor done

DYNINT

Indicates that this packet caused an immediate interrupt via dynamic interrupt moderation.

CRCV

Hardware speculatively found a valid CRC-32. It is up to the software device driver to determine the validity of this indication of a correct CRC-32.

PIF

Hardware supplies the PIF field to expedite software processing of packets. Software must examine any packet with PIF set to determine whether to accept the packet. If PIF is clear, then the packet is known to be for this station so software need not look at the packet contents. In general, packets passing only the Multicast Vector (MTA) but not any of the MAC address exact filters (RAH, RAL) has PIF set. There are considerations that has PIF set:

- In non-promiscuous multicast mode (MCSTCTRL.MPE = 0b) or ignore broadcast mode (FCTRL.BAM = 0b), the PIF bit is set if a packet matches inexact filter (or imperfect – MTA) but not matching any exact (RAH, RAL) filter.

EOP

Packets that exceed the receive buffer size span multiple receive buffers. EOP indicates whether this is the last buffer for an incoming packet.

DD

Indicates whether hardware is finished with the descriptor. When set along with *EOP*, the received packet is complete in main memory. Software can determine buffer usage by setting the status byte *DD* bit to 0b before making the descriptor available to hardware, and checking it for non-zero content at a later time. For multi-descriptor packets, packet status is provided in the final descriptor of the packet (*EOP* set). If *EOP* is not set for a descriptor, only the *Address*, *Length*, and *DD* bits are valid.

VP

The *VP* field indicates whether the incoming packet's type matches VET and the *VLAN* field is stripped (if the packet is a VLAN (802.1q) type). It's set if the packet type matches VET and VLNCTRL.VME is set.

IPCS, L4CS, UDPCS: Bit descriptions are listed in the following table:

Table 3-55. IPCS, L4CS, 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 active and TCP checksum offload. Pass/fail indication is provided in the *Error* field – IPE and TCPE. See PKTTYPE table for supported packet types.
1b	1b	1b/0b	Hardware provides IPv4 checksum offload if IPCS is active along with UDP checksum offload. Pass/fail indication is provided in the *Error* field – IPE and TCPE. See PKTTYPE table for supported packet types.

See Section 3.5.2.12 for a description of supported packet types for receive checksum offloading. IPv6 packets do not have the *IPCS* bit set, but might have the *L4CS* bit set if the 82598 recognized the TCP or UDP packet.

Extended Error Field

Extended Error (12)

- IPE (bit 11) – IPv4 checksum error
- TCPE (bit 10) – TCP/UDP checksum error
- USE (bit 9) – Undersize error – Undersized packet received
- OSE (bit 8) – Oversize error – Oversized packet received
- PE (bit 7) – Packet error – Illegal symbol or error symbol in the middle of a received packet
- Reserved (bit 6) – Reserved
- LE (bit 5) – Length error
- CE (bit 4) – CRC error
- HBO (bit 3) – Header buffer overflow (the header is bigger than the header buffer)
- Reserved (bits 2:0) – Reserved

RXE

Indicates that a data error occurred during the packet reception. A data error refers to the reception of a /FE/ code from the XGMII interface which eventually causes CRC error detection (*CE* bit). This bit is valid only when the *EOP* and *DD* bits are set and are not set in descriptors unless FCTRL.SBP (store-bad-packets) is set. The RXE bit can also be set if a parity error was discovered in the packet buffer while reading this packet. In this case, RXE can be set even if FCTRL.SBP is not set.

IPE

Indicates that the IPv4 header checksum is incorrect.

TCPE

Indicates that the TCP or UDP checksum is incorrect.

The IP and TCP checksum error bits 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.

CE

Indicates an Ethernet CRC error was detected. This bit is valid only when the *EOP* and *DD* bits are set and are not set in descriptors unless FCTRL.SBP (store-bad-packets) is set.

LE

Indicates an Ethernet L2 length error was detected. This bit is valid only when the *EOP* and *DD* bits are set and are not set in descriptors unless FCTRL.SBP (store-bad-packets) is set.

HBO (Header Buffer Overflow)

1. In Header split mode, when SRRCTL BSIZEHEADER is smaller than HDR_BUF_LEN, then HBO is set to 1b. In this case, the header is not split. Instead, the header resides within the host packet buffer. If *SPH* is set, then the HDR_BUF_LEN field is still valid and equal to the calculated size of the header. However, the header is not copied into the header buffer.
2. *HBO* should be ignored each time *SPH* is not set to 1b.

Note: Most error information appears only when the *SBP* bit (FCTRL.SBP) is set and a bad packet is received.

PKT_Buf_Length (16)

Number of bytes exists 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 still reflects the size of the packet, although no data is written to the data buffer. If SRRCTL.DESC_TYPE = 2 (advanced descriptor header splitting) and the buffer is not split because the header is bigger than the allocated header buffer, this field reflects the size of the data written to the data buffer (header + data).

VLAN Tag (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 VLNCTRL.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.

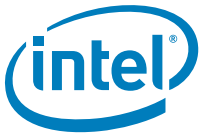


Table 3-56. VLAN Tag Field Layout (for 802.1q Packet)

15 13	12	11 0
Priority	CFI	VLAN

Priority and CFI are part of 803.1q specifications.

3.5.2.5 Receive UDP Fragmentation Checksum

The 82598 might provide Receive fragmented UDP checksum offload. The following setup should be made to enable this mode:

- The RXCSUM.PCSD bit should be cleared. The *Fragment Checksum* and *IP Identification* fields are mutually exclusive with the RSS hash. When the PCSD bit is cleared, the *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 following table lists the outcome descriptor fields for the following incoming packets types.

Table 3-57. Descriptor Fields for Incoming Packet Types

Incoming Packet Type	Fragment Checksum	UDPV	UDPCS/L4CS
Non IPv4 packet	0b	0b	0b/0b for UDP 0b/1b for TCP for IPv6 not fragmented packet else 0b/0b
Non fragmented IPv4 packet	Same as above	0b	Depends on transport header UDP: 1b/1b TCP: 0b/1b
Fragmented IPv4, when not first fragment	The unadjusted 1's complement checksum of the IP payload	0b	1b/0b
Fragmented IPv4 with protocol = UDP, first fragment (UDP protocol present)	Same as above	1b if the UDP header checksum is valid (not 0b)	1b/0b

Note: When the software device driver computes the 16-bit 1's complement sum on the incoming packets of the UDP fragments, it should expect a value of 0xFFFF.

3.5.2.6 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 sections briefly describe the descriptor fetch algorithm and the software control provided.

When the on-chip buffer is empty, a fetch happens as soon as any descriptors are made available (host writes to the tail pointer). When the on-chip buffer is nearly empty (RXDCTL.PTHRESH), a prefetch is performed each time enough valid descriptors (RXDCTL.HTHRESH) are available in host memory and no other PCIe 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 82598 might elect to perform a fetch that is not a multiple of cache line size. The 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 most efficient in the cases where it has fallen behind software.

Note: The 82598 NEVER fetches descriptors beyond the descriptor TAIL pointer.

3.5.2.7 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. Descriptor write backs accumulate and are opportunistically written out in cache line-oriented chunks, under the following scenarios:

- RXDCTL.WTHRESH descriptors have been used (the specified maximum threshold of unwritten used descriptors has been reached)
- The receive timer expires (ITR)
- Dynamic interrupt moderation (immediate bit indicating ITR should be overwritten)

When the number of descriptors specified by RXDCTL.WTHRESH have been used, they are written back, regardless of cache line alignment. It is therefore recommended that WTHRESH be a multiple of cache line size. When the receive timer (ITR) expires, all used descriptors are forced to be written back prior to initiating the interrupt, for consistency.

When the 82598 does a partial cache line write-back, it attempts to recover to cache-line alignment on the next write-back.

Note: Software can determine if a descriptor has been used for packet reception by checking the DD bit of the descriptor written back. Software should not use the Receive Head register as an indication to the descriptor usage by hardware.

3.5.2.8 Receive Descriptor Queue Structure

Figure 3-23 shows the structure of each of the receive descriptor rings with each ring using a contiguous memory space. Hardware maintains internal circular queues of 64 descriptors (per queue) to hold the descriptors that were fetched from the software ring. The hardware 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.

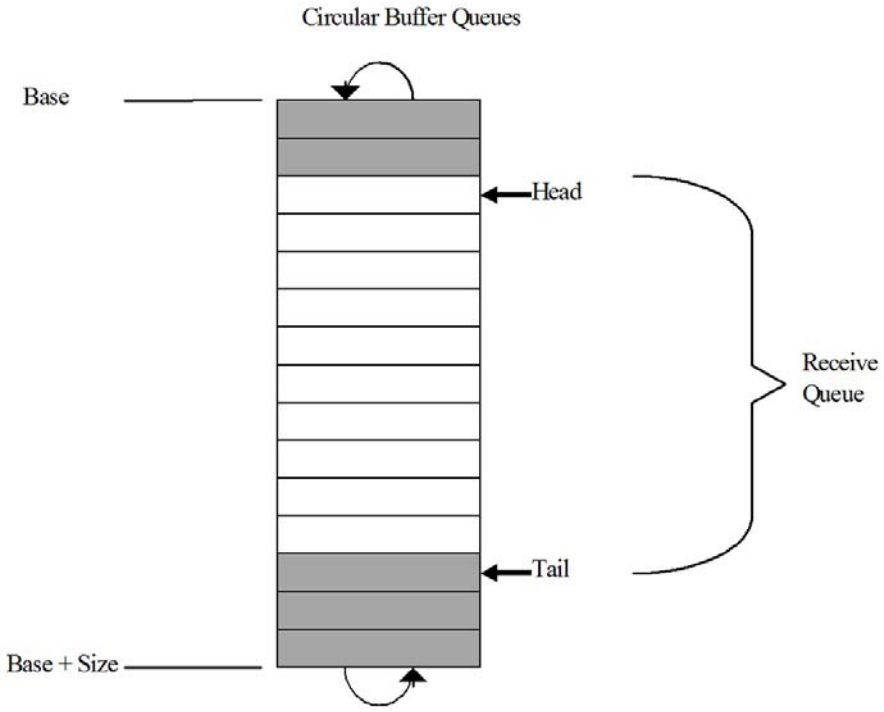


Figure 3-23. Receive Descriptor Ring Structure

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 to16-byte blocks of memory. Shaded boxes in the figure represent descriptors that have stored incoming packets but have not yet been recognized by software. Software can determine if a receive buffer is valid by reading descriptors in memory rather than by IO reads. Any descriptor with a non-zero status byte 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. At the completion of the descriptor write-back operation, this pointer is incremented by the number of descriptors written back. Hardware owns all descriptors between [head .. tail]. Any descriptor not in this range is owned by software.

The receive descriptor rings are described by the following registers:

- Receive Descriptor Base Address registers (RDBA) – 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. Hardware ignores the lower four bits.
- Receive Descriptor Length registers (RDLEN) – 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 registers (RDH) – This register holds a value that is an offset from the base and indicates the in-progress descriptor. There can be up to 32K-8 descriptors in the circular buffer. Hardware maintains a shadow copy that includes those descriptors completed but not yet stored in memory.
- Receive Descriptor Tail registers (RDT) – This register holds a value that is an offset from the base and identifies the location beyond the last descriptor hardware can process. This is the location where software writes the first new descriptor.

If software statically allocates buffers, and uses a memory read to check for completed descriptors, it needs to zero the status byte in the descriptor to make it ready for re-use by hardware. This is not a hardware requirement, but is necessary for performing an in-memory scan.

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

3.5.2.9 Header Splitting and Replication

3.5.2.9.1 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 3.5.6).

The packet (header + payload) is stored in memory through a (optionally) no-snoop transaction. Later, the data movement engine moves the payload from the driver space to the application memory.

The 82598 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
- Advanced mode, split: always use header buffer: Advanced descriptors are in use; header and payload are split to different buffers. If no split is done, the first part of the packet is stored in the header buffer
- Advanced mode, replication: Advanced descriptors are in use; header is replicated in a separate buffer, and also in the 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.

3.5.2.9.2 Description

In Figure 3-24, the header splitting with header replication is described.

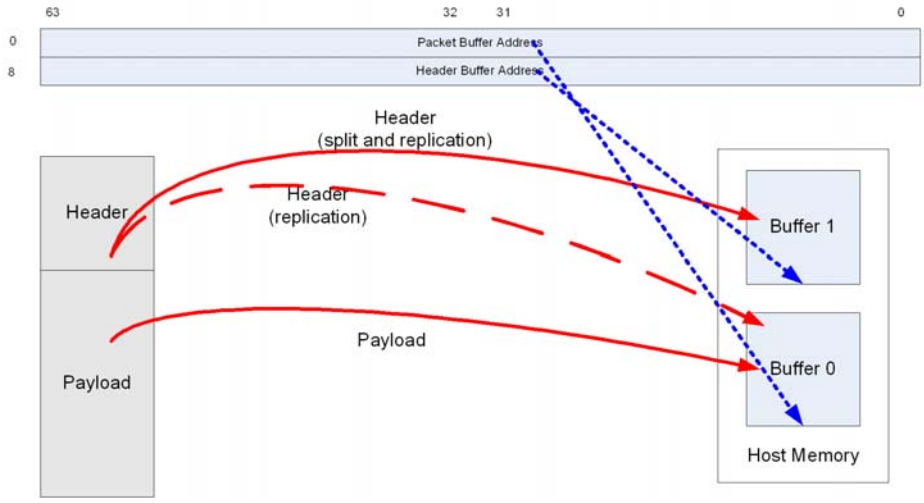


Figure 3-24. Header Splitting with Replicated Header

The physical address of each buffer is written in the *Buffer Addresses* fields:

- 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 case of 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 sizes of these buffers are statically defined in the SRRCTL[n] registers:

- The BSIZEPACKET field defines the size of the buffer for the received packet.
- The BSIZEHEADER field defines the size of the buffer for the received header. If header split or header replication are enabled, this field must be configured to a non-zero value. The amount of data written into the header buffer is different in header split and header replication modes:
  - Header split – the 82598 only writes the header portion into the header buffer. The header size is determined by the options enabled in the PSRTYPE registers.
  - Header replication – the 82598 writes beyond the header up to the full size of the header buffer (or to the size of the packet, if smaller).

The 82598 uses the packet replication or splitting feature when the SRRCTL[n].DESCTYPE > one.

When header split or header replication is selected, the packet is split (or replicated) only on selected types of packets. A bit exists for each option in PSRTYPE[n] registers, so several options can be used in conjunction. If one or more bits are set, the splitting (or replication) is performed for the corresponding packet type. See Section 3.5.2.4 for details on the possible headers type supported.

The following table lists the behavior of the 82598 in the different modes:

Table 3-58. Mode Behavior for 82598

DESCTYPE	Condition	SPH	HBO	PKT_LEN	HDR_LEN	Copy
Split	1. Header can't be decoded	0b	0b	Min (packet length, buffer size)	N/A	Header + Payload → Packet buffer
	2. Header <= BSIZEHEADER	1b	0b	Min (payload length, buffer size)[3]	Header size	Header → Header buffer Payload → Packet buffer
	3. Header > BSIZEHEADER	1b	1b	Min (packet length, buffer size)	Header size[6]	Header + Payload → Packet buffer
Split – always use header buffer	1. Packet length <= BSIZEHEADER	0b	0b	0b	Packet length	Header + Payload → Header buffer
	2. Header can't be Decoded (Packet length > BSIZEHEADER)	0b	0b	Min (packet length – BSIZEHEADER, data buffer size)	BSIZEHEADER	Header + Payload → Header + Packet buffers[4]
	3. Header <= BSIZEHEADER	1b	0b	Min (payload length, data Buffer size)	Header Size	Header → Header buffer Payload → Packet buffer
	4. Header > BSIZEHEADER	1b	1b	Min (packet length – BSIZEHEADER, data buffer size)	Header Size[6]	Header + Payload → Header + Packet buffer
Replicate	1. Header + Payload <= BSIZEHEADER	0b/ 1b	0b	Packet length	Header size, N/A[5]	Header + Payload → Header buffer
	2. Header + Payload > BSIZEHEADER	0b/ 1b	0b/ 1b[2]	Min (packet length, buffer size)	Header size, N/A[5]	(Header + Payload)(partial[1]) → Header buffer Header + Payload → Packet buffer

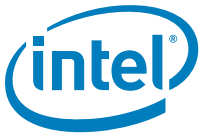
Notes:
1. Partial means up to BSIZEHEADER
2. HBO is 1b if the Header size is bigger than BSIZEHEADER and zero otherwise.
3. In a header only packet (such as TCP ACK packet), the PKT_LEN is zero.
4. If the packet spans more than one descriptor, only the header buffer of the first descriptor is used.
5. If SPH = 0b, then the header size is not relevant. In any case, the HDR_LEN doesn't reflect the actual data size stored in the Header buffer.
6. The HDR_LEN doesn't reflect the actual data size stored in the header buffer. It reflects the header size determined by the parser.

Note: If SRRCTL#.NSE is set, All buffers' addresses in a packet descriptor must be word aligned.

Packet header cannot 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. If header split mode (SRRCTL.DESCTYPE = 010b), a packet with a header larger than the header buffer is not split.

3.5.2.10 Receive-Side Scaling (RSS)

RSS is a mechanism to post each received packet into one of several descriptor queues. Software potentially assigns each queue to a different processor, therefore sharing the load of packet processing among several processors.



As described in Section 3.5.2, the 82598 uses RSS as one ingredient in its packet assignment policy (the other is VMDq). The RSS output is a 4-bit index or a pair of 4-bit indices. The 82598's global assignment uses these bits (or only some of the LSBs) as part of the queue policy.

RSS is enabled in the MRQC register. RSS status field in the descriptor write-back is enabled when the RXCSUM.PCSD bit is set (Fragment Checksum is disabled). RSS is therefore mutually exclusive with UDP fragmentation. Also, support for RSS is not provided when legacy receive descriptor format is used.

When RSS is enabled, the 82598 provides software with the following information, required by Microsoft* RSS or provided for software 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 3-25 shows the process of computing an RSS output:

1. The receive packet is parsed into the header fields used by the hash operation (IP addresses, TCP port, etc.)
2. A hash calculation is performed. The 82598 supports a single hash function, as defined by Microsoft* RSS. The 82598 therefore 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 LSBs of the hash result are used as an index into a 128-entry indirection table. Each entry provides a 4-bit RSS output index or a pair of 4 bit indices.

When RSS is disabled, packets are assigned an RSS output index = 0b. 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 = 0b. The 32-bit tag (normally a result of the hash function) equals 0b.

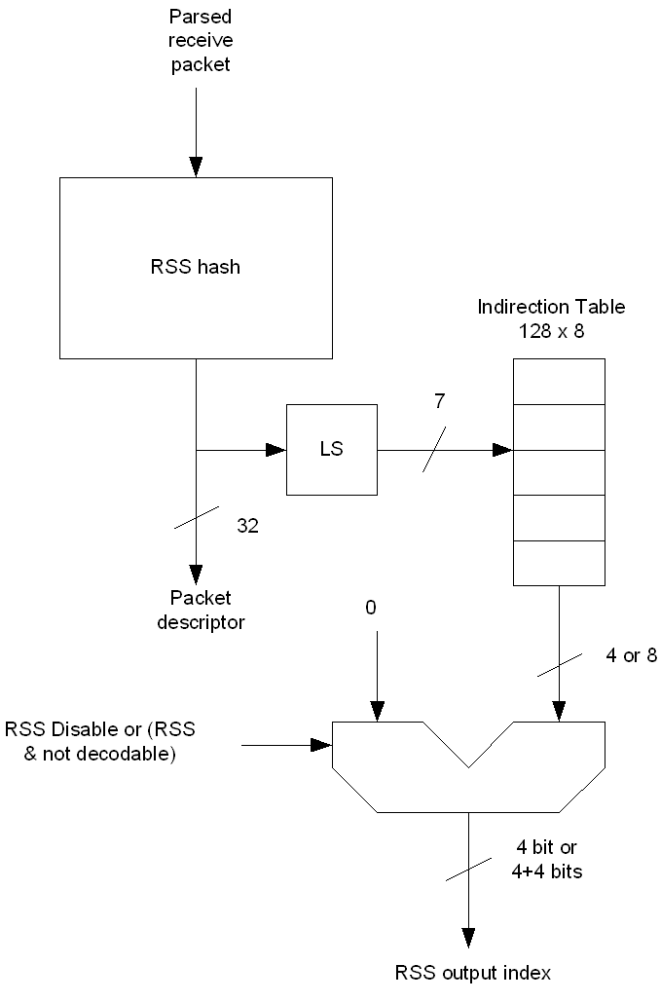


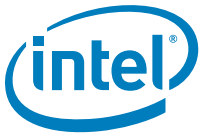
Figure 3-25. RSS Block Diagram

3.5.2.10.1 RSS Hash Function

Section 3.5.2.10.1 provides a verification suite used to validate that the hash function is computed according to Microsoft* nomenclature.

The 82598 hash function follows the Microsoft* definition. A single hash function is defined with several variations for the following cases:

- TcpIPv4 – the 82598 parses the packet to identify an IPv4 packet containing a TCP segment. If the packet is not an IPv4 packet containing a TCP segment, receive-side-scaling is not done for the packet.
- IPv4 – the 82598 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 82598 parses the packet to identify an IPv6 packet containing a TCP segment. If the packet is not an IPv6 packet containing a TCP segment, RSS is not done for the packet.



- TcpIPv6Ex – the 82598 parses the packet to identify an IPv6 packet containing a TCP segment with extensions. 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).
- IPv6 – the 82598 parses the packet to identify an IPv6 packet. If the packet is not an IPv6 packet, RSS is not done for the packet.

The following additional cases are not part of the Microsoft* RSS specification:

- UdpIPv4 – the 82598 parses the packet to identify a packet with UDP over IPv4
- UdpIPv6 – the 82598 parses the packet to identify a packet with UDP over IPv6
- UdpIPv6Ex – the 82598 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 (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. Use the TcpIPv4 function.
2. Use IPv4_UDP function.
3. Use the IPv4 function.

IPv6 packet:

1. If TcpIPv6Ex is enabled, use the TcpIPv6Ex function or if TcpIPv6 is enabled, use the TcpIPv6 function.
2. If UdpIPv6Ex is enabled, use UdpIPv6Ex function or if UpdIPv6 is enabled, use UdpIPv6 function.

The following combinations are currently supported:

- Any combination of IPv4, TcpIPv4, and UdpIPv4.
- And/or
- Any combination of either IPv6, TcpIPv6, and UdpIPv6, TcpIPv6Ex, and UdpIPv6Ex.

When a packet cannot be parsed by the previously stated 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 following hash functions:

- Ordering is little endian in both bytes and bits. For example, the IP address 161.142.100.80 translates into 0xa18e6450 in the signature
- A " ^ " denotes bit-wise XOR operation of same-width vectors
- @x-y denotes bytes x through y (including both of them) of the incoming packet, where byte 0 is the first byte of the IP header. In other words, all byte-offsets as offsets into a packet where the

framing layer header has been stripped out. Therefore, the source IPv4 address is referred to as @12-15, while the destination v4 address is referred to as @16-19.

- @x-y, @v-w denotes concatenation of bytes x-y, followed by bytes v-w, preserving the order in which they occurred in the packet.

All hash function variations (IPv4 and IPv6) follow the same general structure. Specific details for each variation are described in the following section. The hash uses a random secret key of length 320 bits (40 bytes); the key is generally supplied through the RSS Random Key (RSSRK) register.

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 kB, Intel's nomenclature assumes that the array is laid out as follows:

```
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.

3.5.2.10.1.1 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);
```

3.5.2.10.1.2 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);
```

3.5.2.10.1.3 Hash for IPv4 without TCP

Concatenate SourceAddress and DestinationAddress into one single byte-array

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

3.5.2.10.1.4 Hash for IPv6 with TCP

Similar to above:

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

3.5.2.10.1.5 Hash for IPv6 with UDP

Similar to above:

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

3.5.2.10.1.6 Hash for IPv6 without TCP

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

3.5.2.10.2 Indirection Table

The indirection table is a 128-entry structure, indexed by the seven LSBs of the hash function output. Each entry of the table contains the following:

- Bits [3:0] – RSS output index 0
- Bits [7:4] – RSS output index 1 (optional)

System software can 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.

3.5.2.10.3 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
```

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

IPv6

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

Destination Address/Port	Source Address/Port	IPv6 Only	IPv6 with TCP
3ffe:2501:200:1fff::7 (1766)	3ffe:2501:200:3::1 (2794)	0x2cc18cd5	0x40207d3d

ff02::1 (4739)	3ffe:501:8::260:97ff:fe40:efab (14230)	0x0f0c461c	0xdde51bbf
fe80::200:f8ff:fe21:67cf (38024)	3ffe:1900:4545:3:200:f8ff:fe21:67cf (44251)	0x4b61e985	0x02d1feef

3.5.2.11 Receive Queuing for Virtual Machine Devices (VMDq)

Virtual Machine Devices queue (VMDq) is a mechanism to share I/O resources among several consumers. For example, in a virtual system, multiple operating systems are loaded and each executes as though the entire system's resources were at its disposal. However, for the limited number of I/O devices, this presents a problem because each operating system maybe in a separate memory domain and all the data movement and device management has to be done by a Virtual Machine Monitor (VMM). VMM access adds latency and delay to I/O accesses and degrades I/O performance. Virtual Machine Devices (VMDs) are designed to reduce the burden of VMM by making certain functions of an I/O device shared and thus can be accessed directly from each guest operating system or Virtual Machine (VM). The 82598's 64 queues can be accessed by up to 16 VMs if configured properly. When the 82598 is enabled for multiple queue direct access for VMs, it becomes a VMDq device.

Note: Most configuration and resources are shared across queues. System software must resolve any conflicts in configuration between the VMs.

When enabled, VMDq assigns a 4-bit VMDq output index to each received packet. The VMDq output index is used to associate the packet to a receive queue as described in Section 3.5.2. VMDq generates its output index in one of the following ways:

- Receive packets are associated with receive queues based on the packet destination MAC address
- Receive packets are associated with receive queues based on the packet VLAN tag ID

Packets that do not match any of the enabled filters are assigned with the default VMDq output index value. This might include the following cases:

When configured to associate through MAC addresses:

- Promiscuous mode – Promiscuous mode is used by a virtualized environment to support more than 16 VMs, so that the busier VMs are assigned specific queues, while all other VMs share the default queue.
- Broadcast packets
- Multicast packets

3.5.2.11.1 Association Through MAC Address

Each of the 16 MAC address filters can be associated with a VMDq output index. The VIND field in the Receive Address High (RAH) register determines the target queue. Packets that do not match any of the MAC filters (broadcast, promiscuous, etc.) are assigned with the default index value.

Software can program different values to the MAC filters (any bits in RAH or RAL) at any time. The 82598 responds to the change on a packet boundary, but does not guarantee the change to take place at some precise time.

3.5.2.12 Receive Checksum Offloading

The 82598 supports the offloading of four receive checksum calculations:

- Fragment Checksum
- IPv4 Header Checksum

- TCP Checksum
- UDP Checksum

For supported packet/frame types, the entire checksum calculation can be off-loaded to the 82598. The 82598 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, the 82598 calculates the TCP checksum and indicates a pass/fail condition to software via the TCP *Checksum Error* bit (RDESC.TCPE). These error bits are valid when the respective status bits indicate the checksum was calculated for the packet (RDESC.IPCS and RDESC.L4CS respectively). Similarly, if RFCTL.Ipv6_DIS and RFCTL.IP6Xsum_DIS are cleared to zero the 82598 calculates the TCP or UDP checksum for IPv6 packets. It then indicates a pass/fail condition in the TCP/UDP *Checksum Error* bit (RDESC.TCPE).

Supported Frame Types:

- Ethernet II
- Ethernet SNAP

Table 3-59. Supported Receive Checksum Capabilities

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP Checksum Calculation
IP header's protocol field contains a protocol # other than TCP or UDP.	Yes	No
IPv4 + TCP/UDP packets	Yes	Yes
IPv6 + TCP/UDP packets	No (n/a)	Yes
IPv4 Packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
IPv6 packet with next header options: Hop-by-Hop options Destinations options Routing (with LEN 0) Routing (with LEN >0) 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
Packet has TCP or UDP options	Yes	Yes
IPv4 tunnels: IPv4 packet in an IPv4 tunnel IPv6 packet in an IPv4 tunnel	 No Yes (IPv4)	 No No
IPv6 tunnels: IPv4 packet in an IPv6 tunnel IPv6 packet in an IPv6 tunnel	 No No	 No No
Packet is an IPv4 fragment	Yes	UDP checksum assist
Packet Type	**Hardware IP Checksum Calculation**	**Hardware TCP/UDP Checksum Calculation**
Packet is greater than 1522 bytes	Yes	Yes
Packet has 802.3ac tag	Yes	Yes

The previous table 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:

MAC Address Filter

This filter checks the MAC destination address to make sure it is valid (IA match, broadcast, multicast, etc.). The receive configuration settings determine which MAC addresses are accepted. See the various receive control configuration registers such as FCTRL, MCSTCTRL (RTCL.UPE, MCSTCTRL.MPE, FCTRL.BAM), MTA, RAL, and RAH.

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 VLNCTRL.VFE, VLNCTRL.VET, and VFTA.

IPv4 Filter

This filter checks for valid IPv4 headers. The version field is checked for a correct value (four).

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.

IPv6 Filter

This filter checks for valid IPv6 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).

IPv6 Extension Headers

IPv4 and TCP provide header lengths, which allow hardware to easily navigate through these headers on packet reception for calculating checksums 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 it's *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) can be tunneled using IPv6, and thus another IPv6 (or IPv4) header and potentially its extension headers can be found after the extension headers.

The IPv4 *Next Header Type* is at byte offset 9. In IPv6, the first *Next Header Type* is at byte offset 6.

All IPv6 extension headers have the *Next Header Type* in their first 8 bits. Most have the length in the second 8 bits (Offset Byte[1]) as follows:

Table 3-60. Typical IPv6 Extended Header Format (Traditional Representation)

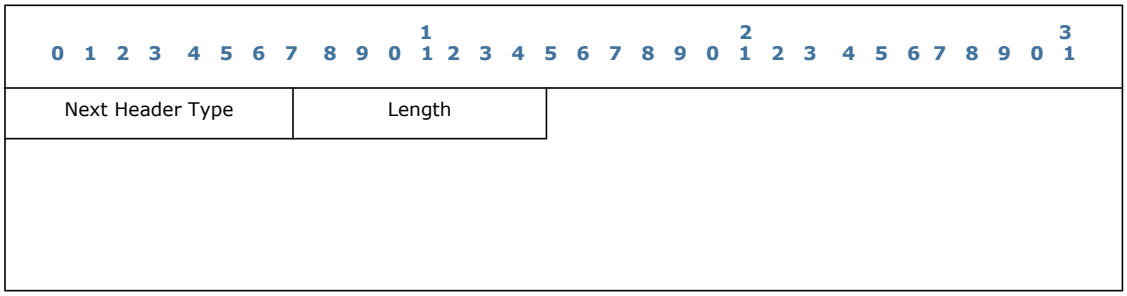


Table 3-61 lists the encodings of the *Next Header Type* field, and information on determining each header type's length. The IPv6 extension headers are not otherwise processed by the 82598 so their details are not covered here.

Table 3-61. Header Type Encodings and Lengths

Header	Next Header Type	Header Length
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]
Header	**Next Header Type**	**Header Length**
Destination Options	60	8+Offset Byte[1]
Routing	43	8+Offset Byte[1]
Fragment	44	Always 8 bytes
Authentication	51	Note 3
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 and Authentication – the 82598 cannot offload packets with this header type.
4. The 82598 hardware acceleration does not support all IPv6 Extension header types, see Table 3-60.

5. The RFCTL.Ipv6_DIS bit must be cleared for this filter to pass.

UDP/TCP Filter

This filter checks for a valid UDP or TCP header. The prototype next header values are 0x11 and 0x06, respectively.

3.5.3 Transmit Functionality

3.5.3.1 Packet Transmission

Output packets are made up of pointer-length pairs constituting a descriptor chain (called descriptor based transmission). Software forms transmit packets by assembling the list of pointer-length pairs, storing this information in the transmit descriptor, and then updating the on-chip transmit tail pointer to the descriptor. The transmit descriptor and buffers are stored in host memory. Hardware transmits the packet only after it has completely fetched all packet data from host memory and deposited it into the on-chip transmit FIFO. This permits TCP or UDP checksum computation, and avoids problems with PCIe under-runs.

Another transmit feature of the 82598 is TCP segmentation. The hardware has the capability to perform packet segmentation on large data buffers off-loaded from the Network Operating System (NOS). This feature is discussed in detail in Section 3.5.3.4.

Transmit tail pointer writes should be to EOP descriptors (the software device driver should not write the tail pointer to a descriptor in the middle of a packet/TSO).

3.5.3.1.1 Transmit Data Storage

Data is stored in buffers pointed to by the descriptors. Alignment of data is on an arbitrary byte boundary with the maximum size per descriptor limited only to the maximum allowed packet size (16 kB). A packet typically consists of two (or more) buffers, one (or more) for the header and one (or more) for the actual data. Each buffer is referred by a different descriptor. Some software implementations copy the header(s) and packet data into one buffer and use only one descriptor per transmitted packet.

3.5.3.2 Transmit Contexts

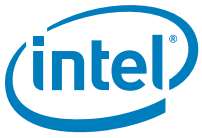
The 82598 provides hardware checksum offload and TCP segmentation facilities. These features enable TCP or UDP packet types to be handled more efficiently by performing additional work in hardware, thus reducing the software overhead associated with preparing these packets for transmission. Part of the parameters used to control these features are handled though contexts.

A context refers to a set of device registers loaded or accessed as a group to provide a particular function. The 82598 supports 256 contexts register sets on-chip. 256 contexts are spread so each eight contexts are related to a separate transmit queue.The transmit queues can contain transmit data descriptors, much like the receive queue, and also transmit context descriptors.

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 contexts that support the transmit checksum offloading and the segmentation features of the 82598.

The 82598 supports one type of transmit context. The extended context is written with a Transmit context descriptor DTYP=2 and this context is always used for transmit data descriptor DTYP=3.

The *IDX* field contains an index to one of eight on-chip per queue contexts. Software must track what context is stored in each *IDX* location.



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 TCP/IP flow. All the packets of this type can be sent using the same context.

Each context controls calculation and insertion of up to two checksums. This portion of the context is referred to as the checksum context. In addition to a checksum context, the segmentation context adds information specific to the segmentation capability. This additional information includes the total size of the MAC header (TDESC.HDRLENMACHDR), the amount of payload data that should be included in each packet (TDESC.MSS), L4 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 and is therefore ignored for context descriptors that do not have the TSE.

Because there is dedicated resources on-chip 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 completely 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 set up once and used for many frames. Only when some other frame type is required would a new context need to be loaded by software using a different index or overwriting an existing context.

This same logic can also be applied to the segmentation context, 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 segmentation context could be used for multiple TCP messages that require hardware segmentation.

3.5.3.3 Transmit Descriptors

The 82598 supports legacy descriptors and advanced descriptors.

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

The legacy descriptors are recognized as such based on the *DEXT* bit.

In addition, the 82598 supports two types of advanced transmit descriptors:

1. Advanced transmit context descriptor, DTYP = 0010b
2. Advanced transmit data descriptor, DTYP = 0011b

Note: DTYP = 0000b and 0001b are reserved values.

The transmit data descriptor points to a block of packet data to be transmitted. The TCP/IP 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.

3.5.3.3.1 Description

3.5.3.3.1.1 Legacy Transmit Descriptor Format

To select legacy mode operation, bit 29 (TDESC.DEXT) should be set to 0b. In this case, the descriptor format is defined as listed in Table 3-62. Address and length must be supplied by software. Bits in the command byte are optional, as are the CSO, and CSS fields.

Table 3-62. Transmit Descriptor (TDESC) Layout – Legacy Mode

	63 48	47 40	39 36	35 32	31 24	23 16	15 0
0	Buffer Address [63:0]						
8	VLAN	CSS	Rsvd	STA	CMD	CSO	Length

Table 3-63. Transmit Descriptor Write Back Format

	63 48	47 40	39 36	35 32	31 24	23 16	15 0
0	Reserved				Reserved		
8	VLAN	CSS	Rsvd	STA	CMD	CSO	Length

Length (16)

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 the supported jumbo frame size 16 kB.

Note: Descriptors with zero length (null descriptors) transfer no data. Null descriptors can appear only between packets and must have their *EOP* bits set.

Checksum Offset and Start – CSO (8) and CSS (8)

A checksum offset (TDESC.CSO) field indicates where, relative to the start of the packet, to insert a TCP checksum if this mode is enabled. A *Checksum Start* (TDESC.CSS) field indicates where to begin computing the checksum. Both CSO and CSS are in units of bytes[1]. These must both be in the range of data provided to the device in the descriptor. This means for short packets that are padded by software, CSS and CSO must be in the range of the unpadded data length, not the eventual padded length (64 bytes).

For an 802.1Q header, the offset values depend on the VLAN insertion enable bit (*VLE*). If they are not set (VLAN tagging included in the packet buffers), the offset values should include the VLAN tagging. If these bits are set (VLAN tagging is taken from the packet descriptor), the offset values should exclude the VLAN tagging.

Hardware does not add the 802.1q Ether Type or the *VLAN* field following the 802.1q Ether Type to the checksum. So for VLAN packets, software can compute the values to back out only on the encapsulated packet rather than on the added fields.

Note: UDP checksum calculation is not supported by the legacy descriptor as the legacy descriptor does not support the translation of a checksum result of 0x0000 to 0xFFFF needed to differentiate between a UDP packet with a checksum of zero and an UDP packet without checksum.

As 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.

1. Even though these are in units of bytes, the checksum calculations of interest typically work on 16-bit words. Hardware does not enforce even-byte alignment.

Note: CSO must be larger than CSS.

Software must compute an offsetting entry-to back out the bytes of the header that should not be included in the TCP checksum-and store it in the position where the hardware computed checksum is to be inserted.

Command Byte – CMD (8)

The CMD byte stores the applicable command and has the fields listed in Table 3-64.

Table 3-64. Transmit Command (TDESC.CMD) Layout

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

- RSV (bit 7) – Reserved
- VLE (bit 6) – VLAN Packet enable
- 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

When EOP is set, it indicates the last descriptor making up the packet. One or many descriptors can be used to form a packet. Hardware inserts a checksum at the offset indicated by the *CSO* field if the *Insert Checksum* bit (IC) is set. Checksum calculations are for the entire packet starting at the byte indicated by the *CSS* field. A value of 0b corresponds to the first byte in the packet. *CSS* must be set in the first descriptor for a packet. In addition, *IC* is ignored if *CSO* or *CSS* are out of range. This occurs if (*CSS* >/= length) OR (*CSO* >/= length = one).

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 do this. If *DD* is set, the descriptor has been processed.

Note: The VLE, IFCS, CSO, and IC fields should be set in the first descriptor for each packet transmitted.

IFCS

When set, the hardware appends the MAC FCS at the end of the packet. When it is cleared, software should calculate the FCS for proper CRC check. There are several cases in which software must set IFCS as follows:

- Transmission of short packet while padding is enabled by the HLREG0.TXPADEN 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

- Large send or TCP/IP checksum offload using context descriptor

VLE indicates that the packet is a VLAN packet (hardware should add the VLAN Ether type and an 802.1q VLAN tag to the packet).

Table 3-65. VLAN Tag Insertion Decision Table for VLAN Mode Enabled

VLE	Action
0	Send generic Ethernet packet.
1	Send 802.1Q packet; the Ethernet Type field comes from the VET register and the VLAN data comes from the VLAN field of the TX descriptor.

Rsvd – Reserved (4)

Status – STA (4)

Table 3-66. Transmit Status (TDESC.STA) Layout

3	2	1	0
Reserved			DD

DD (bit 0) – Descriptor Done Status

This bit provides 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.

When head write-back is enabled, the descriptor write-back is not (with *RS* set).

VLAN (16)

The VLAN field is used to provide the 802.1q/802.1ac tagging information. The *VLAN* field is qualified on the first descriptor of each packet when the *VLE* bit is set to 1b.

Table 3-67. VLAN Field (TDESC.VLAN) Layout

15 13	12	11 0
PRI	CFI	VLAN

3.5.3.3.1.2 Advanced Transmit Context Descriptor

Table 3-68. Transmit Context Descriptor (TDESC) Layout – (Type = 0010b)

	63 48	47 40	39 32	31 16	15 9	8 0
0	RSV			VLAN	MACLEN	IPLEN
8	MSS	L4LEN	IDX	RSV	ADV	DTYP	TUCMD	Reserved
	63 48	47 40	39 36	35 32	31 24	23 20	19 9	8 0

IPLEN (9)

This field holds the value of the IP header length for the IP checksum off-load feature. If an offload is requested, *IPLEN* must be greater than or equal to six, and less than or equal to 511.

MACLEN (7)

This field indicates the length of the MAC header. When an offload is requested (TSE or IXSM or TXSM is set), *MACHDR* must be larger than or equal to 14, and less than or equal to 127.

VLAN (16)

This field contains the 802.1Q VLAN tag to be inserted in the packet during transmission. This VLAN tag is inserted when a packet using this context has its DCMD.VLE bit is set.

TUCMD (11)

- RSV (bit 10-5) – Reserved
- RSV (bit 4) – Reserved
- L4T (bit 3:2) – L4 packet type (00b: UDP; 01b: TCP; 10b, 11b: RSV)
- IPV4(bit 1) – IP packet type: When 1b, IPv4; when 0b, IPv6
- SNAP (bit 0) – SNAP indication

DTYP (4)

This field is always 0010b for this type of descriptor.

ADV (8)

- Reserved (bits 7:6) – Reserved
- DEXT (bit 5) – Descriptor extension (1b for advanced mode)
- Reserved (bits 4:0) – Reserved

IDX (4)

This field holds the index into the hardware context table where this context descriptor is placed. The index is pointing to the per-queue descriptors (eight descriptors).

Note: Because the 82598 supports only eight context descriptors per queue, the MSB is reserved and should be set to 0b.

L4LEN (8)

This field holds the Layer 4 header length. If *TSE* is set, this field is greater than or equal to 12 and less than or equal to 255. Otherwise, this field is ignored.

MSS (16)

This field controls the Maximum Segment Size (MSS). This specifies the maximum TCP payload segment sent per frame, not including any header. The total length of each frame (or section) sent by the TCP segmentation mechanism (excluding Ethernet CRC) is as follows:

1. If *TSE* is set: Total length of an outgoing packet is equal to:

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

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

Software calculates the MSS that is the amount of TCP data that should be used before CRCs are added. Software reduces the MSS sent down to hardware by the maximum amount of bytes that can be added for CRC. The actual number of bytes of TCP data sent out on the wire is greater than this MSS value each time CRCs are added by hardware.

Note: MSS is ignored when DCMD.TSE is not set.

The headers lengths must meet the following:

```
MACLEN + IPLEN + L4LEN < 512
```

Note: MACLEN is augmented by four bytes if VLAN is active.

The context descriptor requires valid data only in the fields used by the specific offload options. The following table describes the required valid fields according to the different offload options.

Table 3-69. Valid Fields by Offload Option

Required Offload			Valid Fields in Context					
TSE	TSXM	IXSM	VLAN	L4LEN	IPLEN MACLEN	MSS	L4T	IPv4
N/A	1	1	VLE	Yes	Yes	Yes	Yes	Yes
N/A	1	1	VLE	Yes	Yes	Yes	Yes	Yes
1	1	1	VLE	Yes	Yes	Yes	Yes	Yes
0	1	X	VLE	No	Yes	No	Yes	Yes
0	0	1	VLE	No	Yes	No	No	Yes
0	0	0	VLE	No	No	No	No	No

3.5.3.3.1.3 Advanced Transmit Data Descriptor

Table 3-70. Advanced Transmit Data Descriptor Read Format

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

Table 3-71. Advanced Transmit Data Descriptor Write-Back Format

0	RSV		
8	RSV	STA	NXTSEQ
	63 36	**35 32**	**31 0**

Address (64)

This field holds the physical address of a data buffer in host memory that contains a portion of a transmit packet.

DTALEN (16)

This field holds the length in bytes of data buffer at the address pointed to by this specific descriptor.

RSV(4)

Reserved

DTYP (4)

0011b for this descriptor type

DCMD (8)

TSE (bit 7) – TCP Segmentation Enable

This field indicates a TCP segmentation request. When *TSE* is set in the first descriptor of a TCP packet, the hardware uses the corresponding context descriptor in order to perform TCP segmentation.

VLE (bit 6) – VLAN Packet Enable

This field indicates that the packet is a VLAN packet (hardware adds the VLAN Ether type and an 802.1q VLAN tag to the packet).

DEXT (bit 5) – Descriptor Extension

This field must be 1b to indicate advanced descriptor format (as opposed to legacy)

RS (bit 3) – Report Status

This field 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.

Note: When the *RS* bit is not used to force write back of descriptors, the 82598 does not write back descriptor or update the head pointer until half of the internal descriptor cache is available for write back (32 descriptors). The software device driver must make sure that it doesn't wait for such a release of those descriptors before handling new ones to the 82598 as it might result is a deadlock situation. To guarantee that this case doesn't occur, a packet should not span more than (host ring size – 31) descriptors.

IFCS (bit 1) – Insert FCS

When this field is set, the 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 as follows:

- Transmission of short packet while padding is enabled by the HLREG0.TXPADEN bit
- Checksum offload is enabled by the either *IC TXSM* or *IXSM* bits in the TDESC.DCMD
- VLAN header insertion enabled by the *VLE* bit in the TDESC.DCMD
- TCP segmentation offload enabled by the *TSE* bit in the TDESC.DCMD

EOP (bit 0) – End of Packet

Packets can span multiple transmit buffers. *EOP* indicates whether this is the last buffer for an incoming packet.

Note: It is recommended that HLREG0.TXPADEN be enabled when *TSE* is true since the last frame can be shorter than 60 bytes – resulting in a bad frame if TXPADEN is disabled. Descriptors with zero length, transfer no data. Even if they have the *RS* bit in the command byte set, the *DD* field in the status word is not written when hardware processes them.

STA (4)

Rsv (bit 3:2) – Reserved

DD (bit 0) – Descriptor Done

IDX (4)

This field holds the index into the hardware context table to indicate which of the eight per-queue contexts should be used for this request.

POPTS (6)

RSV (bit 5) – Reserved

TXSM (bit 1) – Insert TCP/UDP Checksum

When 1b, TCP/UDP checksum is inserted. In this case TUCMD.LP4 indicates whether the checksum is TCP or UDP. When DCMD.TSE is set TXSM must be set to 1b.

IXSM (bit 0) – Insert IP Checksum

This field indicates that IP checksum is inserted. In IPv6 mode, it must be reset to 0b.

If DCMD.TSE is set, and TUCMD.IPV4 is set, IXSM must be set to 1b.

PAYLEN (18)

This field indicates the total length in bytes of the large send packet.

PAYLEN is ignored if TSE is not set.

Note: When a packet spreads over multiple descriptors, all the descriptor fields are only valid in the 1st descriptor of the packet, except for *RS*, which is always checked, and *EOP*, which is always set at last descriptor of the series.

3.5.3.3.2 Transmit Descriptor Structure

The transmit descriptor ring structure is shown in Figure 3-26 each ring uses a contiguous memory space. A pair of hardware registers maintains the transmit descriptor ring in the host memory. New descriptors are added to the ring by software by writing descriptors into the circular buffer memory region and moving the tail pointer associated with that ring. The tail pointer points 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.

Hardware maintains internal circular queues of 64 descriptors per queue to hold the descriptors that were fetched from the software ring. The hardware writes back used descriptors just prior to advancing the head pointer(s).

Descriptors passed to hardware should not be manipulated by software until the head pointer has advanced past them.

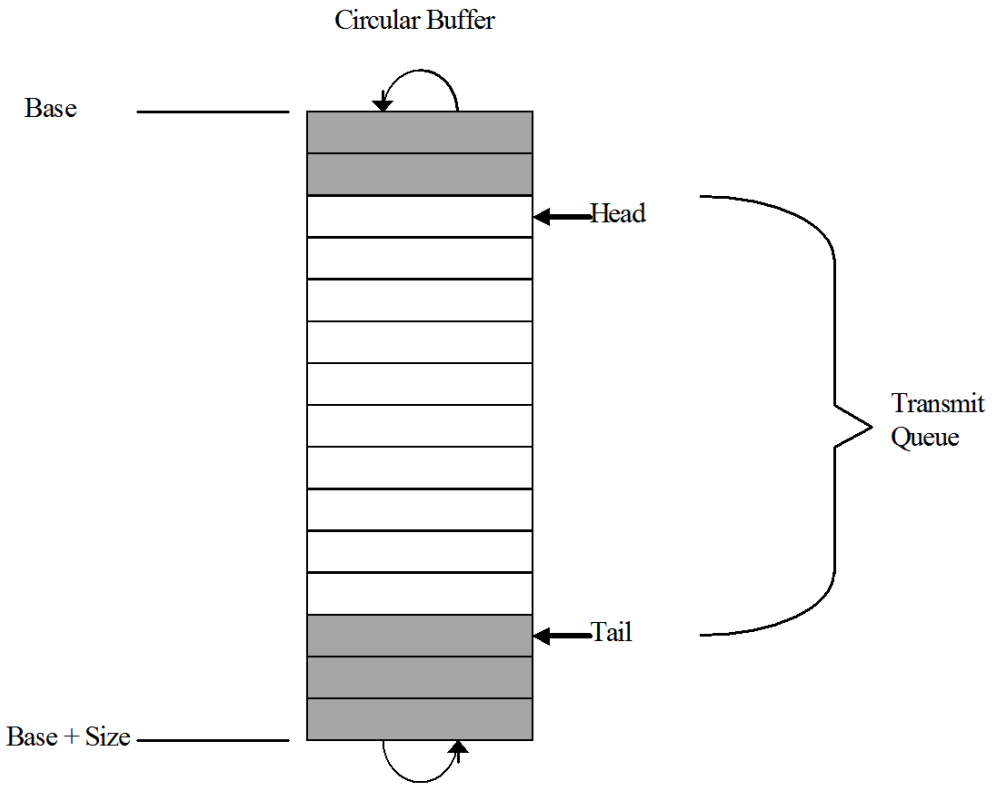


Figure 3-26. Transmit Descriptor Ring Structure

Shaded boxes in Figure 3-26 show descriptors that have been transmitted but not yet reclaimed by software. Reclaiming involves freeing up buffers associated with the descriptors.

The transmit descriptor ring is described by the following registers:

- Transmit Descriptor Base Address (TDBA) register (31:0) – 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 (TDLEN) register (31:0) – This register determines the number of bytes allocated to the circular buffer. This value must be 0b modulo 128.
- Transmit Descriptor Head (TDH) register (31:0) – This register holds a value that is an offset from the base and indicates the in-progress descriptor. There can be up to 32K-8 descriptors in the circular buffer. Reading this register returns the value of head corresponding to descriptors already loaded in the output FIFO.

- Transmit Descriptor Tail (TDT) register (31:0) – This register holds a value that 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 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 64 kB-8 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 beyond the last valid descriptor. Software reads the head register to determine which packets those logically before the head have been transferred to the on-chip FIFO or transmitted.

All the registers controlling the descriptor rings behaviors 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 setting the *RS* bit in the transmit descriptor command field and checking the transmit descriptor *DD* bit in memory.

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.

The process of checking for completed packets consists of one of the following:

- Scan memory.
- Read the hardware head register. All packets up to but **excluding** the one pointed to by head have been sent or buffered and can be reclaimed.
- Issue an interrupt. An interrupt condition is generated each time a packet was transmitted or received and a descriptor was write-back or transmit queue goes empty (EICR.RTxQ[0-19]). This interrupt can either be enabled or masked.

3.5.3.3.3 Transmit Descriptor Fetching

The descriptor processing strategy for transmit descriptors is essentially the same as for receive descriptors except that a different set of thresholds are used.

When the on-chip buffer is empty, a fetch happens as soon as any descriptors are made available (host writes to the tail pointer). When the on-chip buffer is nearly empty (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 82598 might elect to perform a fetch that is not a multiple of cache line size. The 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: Software tail updates should be done at packet boundaries. For example, the last valid descriptor should have its *EOP* bit set. The last valid descriptor should not be a context descriptor.

The 82598 NEVER fetches descriptors beyond the descriptor tail pointer.

3.5.3.3.4 Transmit Descriptor Write-Back

The descriptor write-back policy for transmit descriptors is similar to that for receive descriptors with a few additional factors.

Descriptors are written back in one of three cases:

- TXDCTL[n].WTHRESH = zero and a descriptor which has *RS* set is ready to be written back
- The corresponding ITR counter has reached zero
- TXDCTL[n].WTHRESH > zero and TXDCTL[n].WTHRESH descriptors have accumulated

For the first condition, write-backs are immediate. This is the default operation.

The other two conditions are only valid if descriptor bursting is enabled. In the second condition, the ITR counter is used to force a 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 final condition, if TXDCTL[n].WTHRESH descriptors are ready for write-back, the write-back is performed.

Another possibility for descriptor write back is to use the transmit completion head write-back as explained in Section 3.5.3.7.

3.5.3.4 TCP Segmentation

Hardware TCP segmentation is one of the off-loading options of the TCP/IP stack. This is often referred to as 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) is unique for each packet of the TCP message, and other fields such as the source IP address is constant for all packets associated with the TCP message.

CRC appending (HLREG0.TXCRCEN) must be enabled in TCP segmentation mode because CRC is inserted by hardware. Padding (HLREG0.TXPADEN) must be enabled in TCP segmentation mode, since the last frame might be shorter than 60 bytes – resulting in a bad frame if TXPADEN is disabled.

The offloading of these mechanisms to the software device driver and the 82598 saves significant CPU cycles. The software device driver shares the additional tasks to support these options with the 82598.

Although the 82598's TCP segmentation offload implementation was specifically designed to take advantage of Microsoft's* TCP Segmentation Offload (TSO) feature, the hardware implementation was made generic enough so that it could also be used to segment traffic from other protocols. For example, this feature could be used any time it is desirable for hardware to segment a large block of data for transmission into multiple packets that contain the same generic header.

3.5.3.4.1 Assumptions

The following assumptions apply to the TCP segmentation implementation in the 82598:

- The *RS* bit operation is not changed. Interrupts are set after data in the buffers pointed to by individual descriptors are transferred to hardware.

3.5.3.4.2 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 device driver and commands the driver 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 NOS when the hardware has completed the DMA transfer of the frame (indicated by an interrupt).

The transmission process for the 82598 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.

The hardware 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 the hardware has completed the DMA transfer of the entire data block (indicated by an interrupt).

3.5.3.4.2.1 TCP Segmentation Performance

Performance improvements for a hardware implementation of TCP segmentation offload include:

- The stack does not need to partition the block to fit the 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 software device driver only once per block transfer, instead of once per frame.
- Larger PCI bursts are used which improves bus efficiency (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.

3.5.3.4.3 Packet Format

A TCP message can be as large as 256 kB and is generally fragmented across multiple pages in host memory. The 82598 partitions the data packet into standard Ethernet frames prior to transmission. The 82598 supports calculating the Ethernet, IP, TCP, and UDP headers (including checksum) on a frame-by-frame basis.

Table 3-72. TCP/IP Packet Format

Ethernet	IPv4/IPv6	TCP/UDP	DATA	FCS

Frame formats supported by the 82598 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 is handled by hardware.

Note: IP tunneled packets are not supported for offloading under large send operation.

The 82598 does not support full offload of ECN bits in the TCP header via TCP segmentation to resolve when ever an ECN response is needed software can send the first segment with the *CWR* bit set and the rest of the segments offloaded as TSO with the *CWR* bit clear.

3.5.3.4.4 TCP Segmentation Indication

Software indicates a TCP segmentation transmission context to the hardware by setting up a TCP/IP context transmit descriptor (see Section 3.5.3.3). The purpose of this descriptor is to provide information to the hardware to be used during the TCP segmentation offload process.

Setting the *TSE* bit in the DCMD field to 1b indicates that this descriptor refers to the TCP segmentation context (as opposed to the normal checksum offloading context). This causes the checksum offloading, packet length, header length, and maximum segment size parameters to be loaded from the descriptor into the 82598.

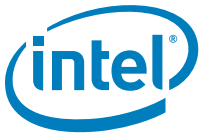
The TCP segmentation prototype header is taken from the packet data itself. Software must identity the type of packet that is being sent (IPv4/IPv6, TCP/UDP, other), calculate appropriate checksum offloading values for the desired checksums, and calculate the length of the header which is prepended. The header can 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 software device driver for modification in constructing the prototype header.

IP Header

- Length should be set to zero



For IPv4 headers

- *Identification* Field should be set as appropriate for first packet of send (if not already)
- 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 checksum as follows (there is a more detailed discussion of this in Section 3.5.3.4.5:

Table 3-73. TCP Partial Pseudo-Header Checksum for IPv4

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

Table 3-74. TCP Partial Pseudo-Header Checksum for IPv6

IPv6 Source Address	
IPv6 Final Destination Address	
Zero	
Zero	Next Header

UDP Header

- Checksum should be set as in TCP header previously described.

The 82598's DMA function fetches the Ethernet, IP, and TCP/UDP prototype header information from the initial descriptor(s) and saves them (on-chip) for individual packet header generation. The following sections describe the updating process performed by the hardware for each frame sent using the TCP segmentation capability.

3.5.3.4.5 IP and TCP/UDP Headers

This section outlines the format and content for the IP, TCP, and UDP headers. The 82598 requires baseline information from the device driver in order to construct the appropriate header information during the segmentation process. Note that header fields that are modified by the 82598 are highlighted in the figures that follow.

Note: IPv4 requires the use of a checksum for the header and 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 (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.

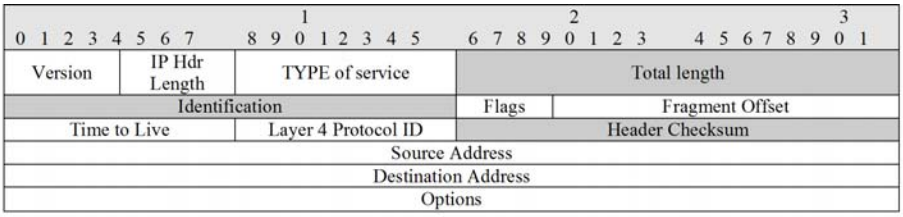


Figure 3-27. IPv4 Header (Traditional Representation)

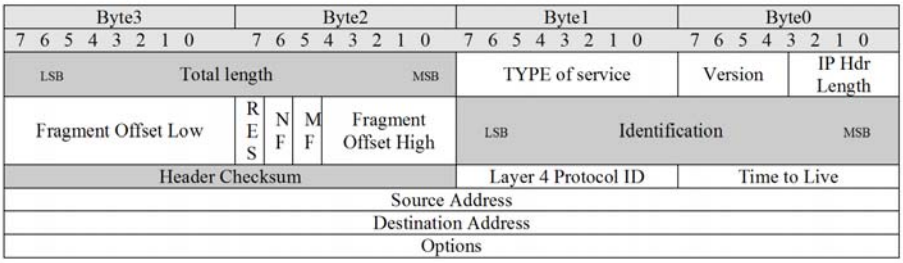


Figure 3-28. IPv4 Header (Little Endian Order)

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 82598 does TCP segmentation, not IP fragmentation. IP fragmentation might occur in transit through a network's infrastructure.

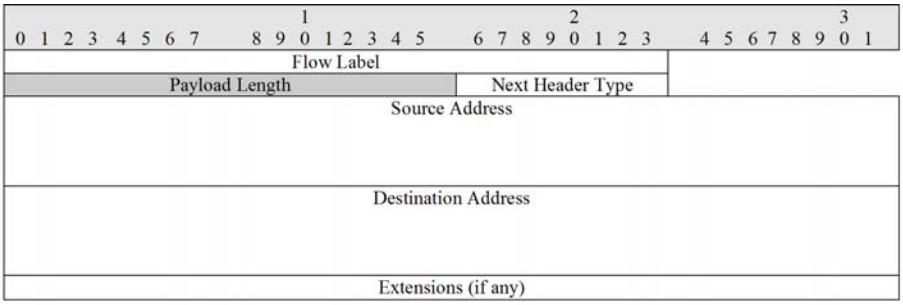


Figure 3-29. IPv6 Header (Traditional Representation)

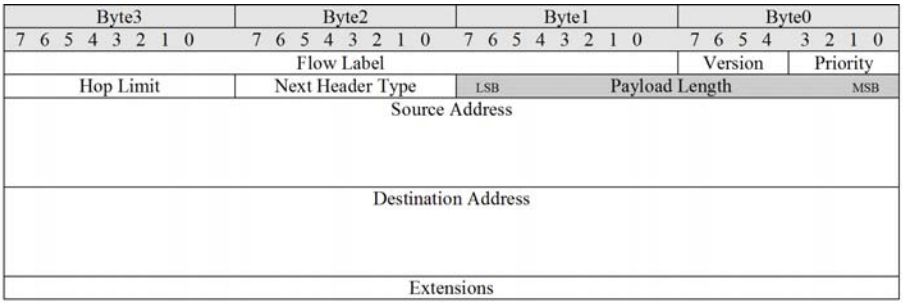


Figure 3-30. IPv6 Header (Little Endian Order)

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 3.5.3.4.6.

Note: TCP requires the use of checksum; optional for UDP.

The TCP header is first shown in the traditional (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.

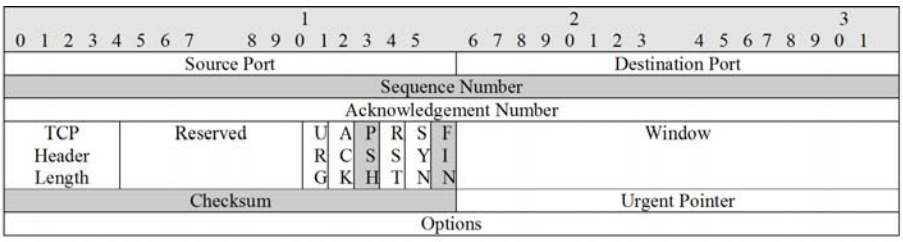


Figure 3-31. TCP Header (Traditional Representation)

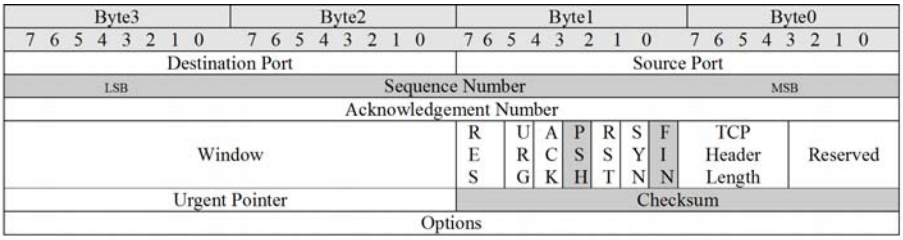


Figure 3-32. TCP Header (Little Endian)

The TCP header is always a multiple of 32-bit words. TCP options can occupy space at the end of the TCP header and are a multiple of eight 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 Figure 3-33). 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, 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 can 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 3.5.3.4.7, IP/TCP/UDP Header Updating.

Payload is normally MSS except for the last packet where it represents the remainder of the payload.

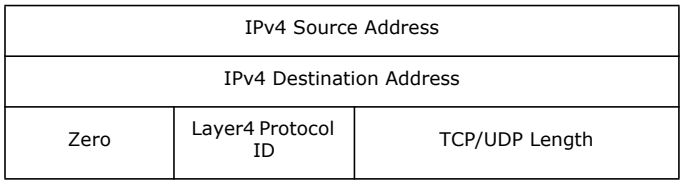


Figure 3-33. TCP/UDP Pseudo Header Content for IPv4 (Traditional Representation)

The Layer 4 Protocol ID value in the pseudo-header identifies the upper-layer protocol (such as, 6 for TCP or 17 for UDP).

IPv6 Source Address	
IPv6 Final Destination Address	
TCP/UDP Packet Length	
Zero	Next Header

Figure 3-34. TCP/UDP Pseudo Header Content for IPv6 (Traditional Representation)

Note: From the RFC2460 specification:

- 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 is in the last element of the routing header; at the recipient(s), that address is in the *Destination Address* field of the IPv6 header.
- The next header value in the pseudo-header identifies the upper-layer protocol (such as, 6 for TCP or 17 for UDP). It differs 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 (TCP header plus TCP data). Some upper-layer protocols carry their own length information (such as *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:

Table 3-75. 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]			

- Next Header – 8-bit selector. 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 eight 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, for example, 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.

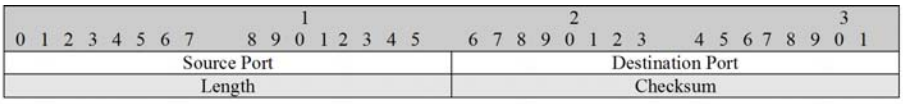


Figure 3-35. UDP Header (Traditional Representation)

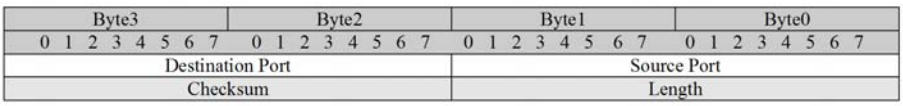


Figure 3-36. UDP Header (Little Endian Order)

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 previously discussed). This checksum procedure is the same as is used in TCP.

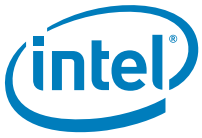
Unlike the TCP checksum, the UDP checksum is optional (for IPv4). Software must set the TXSM bit in the TCP/IP Context Transmit Descriptor to indicate that a UDP checksum should be inserted. Hardware does not overwrite the UDP checksum unless the TXSM bit is set.

3.5.3.4.6 Transmit Checksum Offloading with TCP Segmentation

The 82598 supports checksum off-loading as a component of the TCP segmentation offload feature and as a standalone capability. Section 3.5.3.4.8 describes the interface for controlling the checksum off-loading feature. This section describes the feature as it relates to TCP segmentation.

The 82598 supports IP and TCP/UDP header options in the checksum computation for packets that are derived from the TCP segmentation feature.

Note: The 82598 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 software device driver has to compute all but one IP header checksum. The 82598 calculates checksums on the fly on a frame-by-frame basis and inserts the result in the IP/TCP/UDP headers of each frame. The TCP and UDP checksums are a result of performing the checksum on all bytes of the payload and the pseudo header.



Three specific types of checksum are supported by the hardware in the context of the TCP segmentation offload feature:

- IPv4 checksum
- TCP checksum
- UDP checksum

Each packet that is sent via the TCP segmentation offload feature optionally includes the IPv4 checksum and the TCP or UDP checksum.

All checksum calculations use a 16-bit wide one's complement checksum. The checksum word is calculated on the outgoing data. The checksum field is written with the 16-bit one's complement of the one's complement sum of all 16-bit words in the range of CSS to CSE, including the checksum field itself.

Table 3-76. Supported Transmit Checksum Capabilities

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP Checksum Calculation
IPv4 packets	Yes	Yes
IPv6 packets (no IP checksum in IPv6)	NA	Yes
Packet is greater than 1552 bytes	Yes	Yes
Packet has 802.3ac tag	Yes	Yes
Packet has IP option (IP header is longer than 20 bytes)	Yes	Yes
Packet has TCP or UDP options	Yes	Yes
IP header's protocol field contains protocol # other than TCP or UDP.	Yes	No

The following table lists the conditions of when checksum offloading can/should be calculated.

Packet Type	IPv4	TCP/UDP	Reason
Non-TSO	Yes	No	IP raw packet (non-TCP/UDP protocol)
	Yes	Yes	TCP segment or UDP datagram with checksum offload
	No	No	Non-IP packet or checksum not offloaded
TSO	Yes	Yes	For TSO, checksum offload must be done

3.5.3.4.7 IP/TCP/UDP Header Updating

IP/TCP/UDP header is updated for each outgoing frame based on the IP/TCP header prototype which hardware DMA's from the first descriptor(s) and stores on chip. The IP/TCP/UDP headers are fetched from host memory into an on-chip 240 byte header buffer once for each TCP segmentation context (for performance reasons, this header is not fetched again for each additional packet that is derived from the TCP segmentation process). 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 82598.

Note: Software must make PAYLEN and HDRLEN value of context descriptors correct. Otherwise, the failure of TSOs due to either under-run or over-run can cause hardware to send bad packets or even cause TX hardware to hang. The indication of a TSO failure can be checked in the TSTFC statistic register.

3.5.3.4.7.1 TCP/IP/UDP Header for the first Frames

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

IPv4 Header

- IP Total Length = MSS + L4LEN + IPLEN
- IP Checksum

IPv6 Header

- Payload Length = MSS + L4LEN + IPLEN – 0x28 (IP base header length)

TCP Header

- Sequence Number: The value is the sequence number of the first TCP byte in this frame.
- If FIN flag = 1b, it is cleared in the first frame.
- If PSH flag =1b, it is cleared in the first frame.
- TCP Checksum

UDP Header

- UDP length: MSS + L4LEN
- UDP Checksum

3.5.3.4.7.2 TCP/IP/UDP Header for the Subsequent Frames

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). N is the number of frames that have been transmitted.

MAC Header (for SNAP packets)

- Type/Len field = MSS + MACLEN + IPLEN + L4LEN – 14

IPv4 Header

- IP Identification: incremented from last value (wrap around)
- IP Total Length = MSS + L4LEN + IPLEN
- IP Checksum

IPv6 Header

- Payload Length = MSS + L4LEN + IPLEN – 0x28 (IP base header length)

TCP Header

- Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
- If FIN flag = 1b, it is cleared in these frames.
- If PSH flag =1b, it is cleared in these frames.
- TCP Checksum

UDP Header

- UDP Length: MSS + L4LEN
- UDP Checksum

3.5.3.4.7.3 TCP/IP/UDP Header 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

IPv4 Header

- IP total length = last frame payload bytes + L4LEN + IPLEN
- IP identification: incremented from last value (wrap around configurable based on 15-bit width or 16-bit width)
- IP Checksum

IPv6 Header

- Payload length = last frame payload bytes + L4LEN + IPLEN – 0x28 (IP base header length)

TCP Header

- Sequence number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
- If FIN flag = 1b, set it in this last frame
- If PSH flag =1b, set it in this last frame
- TCP Checksum

UDP Header

- UDP length: last frame payload bytes + L4LEN

- UDP Checksum

3.5.3.4.8 IP/TCP/UDP Checksum Offloading

The 82598 performs checksum offloading as an optional part of the TCP/UDP segmentation offload feature.

These specific checksums are supported under TCP segmentation:

- IPv4 checksum
- TCP checksum
- UDP checksum

Checksum offloading may also be performed in a single-send packet.

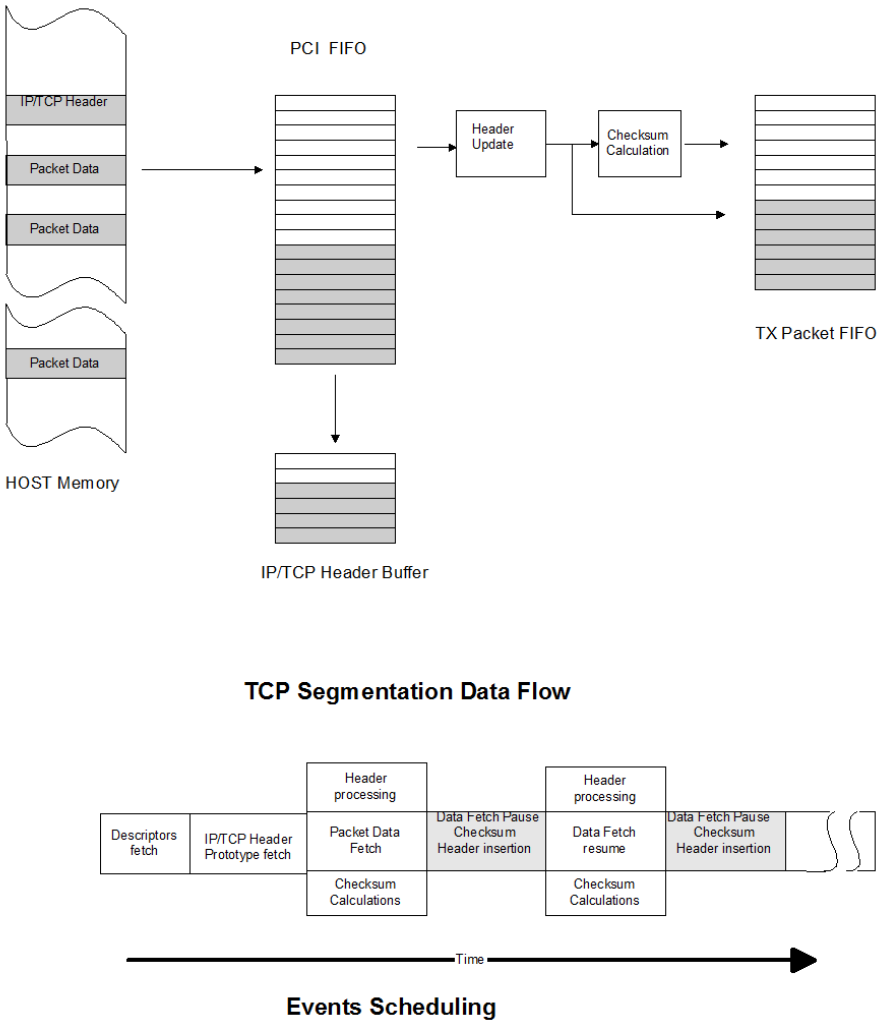


Figure 3-37. Data Flow

3.5.3.5 IP/TCP/UDP Transmit Checksum Offloading in Non-Segmentation Mode

The previous section on TCP segmentation offload describes the IP/TCP/UDP checksum offloading 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 82598's enhanced checksum offload capability, a checksum context must be initialized. For the normal transmit checksum offload feature this is performed by providing the device with a TCP/IP context descriptor. For additional details on contexts, refer to Section 3.5.3.3.2.

Note: Enabling the checksum offloading capability without first initializing the appropriate checksum context leads to unpredictable results.

Note: CRC appending (HLREG0.TXCRCEN) must be enabled in TCP/IP checksum mode, since CRC must be inserted by hardware after the checksums have been calculated.

As mentioned in Section 3.5.3.3, 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.

Each checksum operates independently. Inserting the IP and TCP checksums for each packet are enabled through the transmit data descriptor POPTS.TSXM and POPTS.IXSM fields, respectively.

3.5.3.5.1 IP Checksum

Three fields in the transmit context descriptor set the context of the IP checksum offloading 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.IP=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 transferred 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 14. 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 the maximum allowed value is 511 bytes (the IP checksum should stop after MACLEN+IPLEN. 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.

3.5.3.5.2 TCP Checksum

Three fields in the transmit context descriptor set the context of the TCP checksum offloading feature:

- MACLEN
- IPLEN
- TUCMD.L4T

TUCMD.L4T=1b 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=0 indicates that the packet is UDP and that the 16-bit checksum should be inserted starting at byte offset MACLEN+IPLEN+6.

MACLEN+IPLEN specifies the byte offset from the start of the transferred 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 18/28 for UDP or TCP, respectively. Note that the maximum value for these fields is 127 for MACLEN and 511 for IPLEN. 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.

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.

3.5.3.6 Multiple Transmit Queues

The number of transmit queues is increased to 32 to support multiple CPUs and virtual systems.

3.5.3.6.1 Description

In transmission, each processor sets a queue in the host memory.

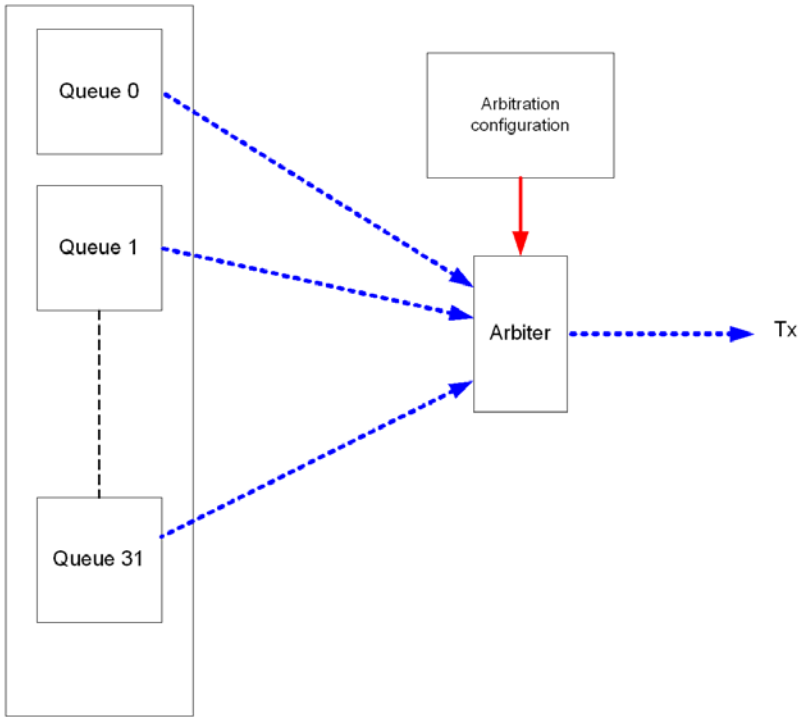


Figure 3-38. Multiple Queues in Transmit

3.5.3.7 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 is complete, 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. To improve the performance of this feature, the software device driver needs to program the DCA registers to configure which CPU is processing each TX queue.

3.5.3.7.1 Description

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

Head write-back occurs if TDWBAL#.Head_WB_En is set for this queue and the *RS* bit is set in the Tx descriptor, following a corresponding data upload into packet buffer.

The software device driver has control on this feature through Tx queue 63:0 write-back address, low and high (thus allowing 64-bit address).

The low register's LSB hold the control bits.

- The Head_WB_En bit enables activation of head write-back. In this case, no descriptor write-back is executed.
- The upper 30 bits of this register hold the lowest 32 bits of the head write-back address, assuming that the two last bits are zero.

The high register holds the high part of the 64-bit address.

The 82598 writes the 32 bits of the queue head register to the address pointed by the TDEWBAH/ TDWBAL registers.

3.5.4 Interrupts

3.5.4.1 Registers

The interrupt logic consists of the registers listed in the following table, plus the registers associated with MSI/MSI-X signaling.

Register	Acronym	Function
Extended Interrupt Cause	EICR	Extended ICR. Records all interrupt causes – an interrupt is signaled when unmasked bits in this register are set.
Extended Interrupt Cause Set	EICS	Enables software to set bits in the Extended Interrupt Cause register.
Extended Interrupt Mask Set/Read	EIMS	Sets or read bits in the extended interrupt mask.
Extended Interrupt Mask Clear	EIMC	Clears bits in the extended interrupt mask.

Extended Interrupt Auto Clear	EIAC	Enables bits in the EICR to be cleared automatically following MSI-X interrupt without a read or write of the EICR.
Extended Interrupt Auto Mask	EIAM	Enables bits in the EIMS to be set and cleared automatically.

Extended Interrupt Cause Registers (EICR)

This register records the interrupt causes to provide to the software information on the interrupt source.

The interrupt causes include:

1. The Rx and Tx queues, each queue can be mapped to one of the 16 interrupt cause bits (RTxQ) available in this register, in non MSI-X mode this mapping is defined by the 82598 software device driver and it uses the same mapping mechanism used in the MSI-X allocation registers (IVAR). See Section 3.5.4.6 for more details on the mapping mechanism.
2. Indication for the TCP timer interrupt.
3. Other bits in this register are the legacy indication of interrupts as the *SDP* bits, management, unrecoverable ECC errors and link status change. There is a specific *Other Cause* bit that is set if one of these bits are set, this bit can be mapped to a specific MSI-X interrupt message.

In MSI-X mode the bits in this register can be configured to auto-clear when the MSI-X interrupt message is sent, in order to minimize driver overhead, and when using MSI-X interrupt signaling. In addition, software can configure the register not to be read-on clear beside *Other Cause* bits if the GPIE.OCD bit is set. When set, only the other causes bits are clear on read – The only case where software reads the EICR in this mode, is if the *Other* interrupt bit is set.

In systems that do not support MSI-X, reading the EICR register clears it's bits or writing 1b's clears the corresponding bits in this register. Most systems have write buffers that minimizes overhead, but this might require a read operation to guarantee that the write has been flushed from posted buffers.

Extended Interrupt Cause Set Register (EICS)

This registers enables triggering an immediate interrupt by software, By writing 1b to bits in EICS the corresponding bits in EICS is set and the relevant EITR is reset (as if the counter was written to zero) if GPIE.EIMEN bit is set. If GPIE.EIMEN bit is not set, than setting the bit does not cause an immediate interrupt, but it waits for the EITR to expire. Used usually to rearm interrupts, software didn't have time to handle in the current interrupt routine.

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 1b and the corresponding interrupt mask bit is 1b. Software blocks asserting 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. Clear and set 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 to a one in the set register and cleared for each bit written in the clear register. Reading the set register (EIMS) returns the current mask register value.

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 EICR is automatically cleared following an interrupt.

When used in conjunction with MSI-X interrupt vector, this feature enables interrupt cause recognition and selective interrupt cause and mask bits reset without requiring software to read the EICR register. As a result, the penalty related to a PCIe read transaction is avoided.

Extended Interrupt Auto Mask Enable register (EIAM)

Each bit in this register enables the setting of the corresponding bit in the EIMC register following a write-to-clear to the EICR register or setting the corresponding bit in the EIMS register following a write-to-set to EICS.

This register is provided in case MSI-X is not used, and therefore auto-clear through EIAC register is not available.

In addition, when in MSI-X mode and GPIE.EIAME is set, software can set the bits of this register to select mask bits that is reset during interrupt processing. In this mode, each bit in this register enables setting of the corresponding bit in EIMC following interrupt generation.

3.5.4.2 Interrupt Moderation

An interrupt is generated upon receiving of incoming packets, as throttled by the EITR registers. There are 16 EITR registers, each one is allocated to a vector of MSI-X.

When an MSI-X interrupt is activated, each active bit in EICR can trigger an interrupt vector. Allocating MSI-X vectors is set by the setting of IVAR[23:0] registers. Following the allocation, the EITR corresponding to the MSI-X vector is tied to the same allocation (EITR0 is allocated to MSI-X[0] and its corresponding interrupts, EITR1 is allocated to MSI-X[1] and its corresponding interrupts etc.).

When MSI-X is not activated, the interrupt moderation is controlled by 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 82598, regardless of network traffic conditions.

The following algorithm to convert the inter-interrupt interval value to the common interrupts/sec performance metric:

Interrupts/sec = $(256 * 10^{-9}\text{sec} * \text{interval})^{-1}$

For example, if the interval is programmed to 500d, the 82598 guarantees the CPU is not interrupted by the 82598 for at least 128 μs from the last interrupt. The maximum observable interrupt rate from the 82598 should not exceed 7813 interrupts/sec.

Inversely, inter-interrupt interval value can be calculated as:

Inter-interrupt interval = $(256 * 10^{-9}\text{sec} * \text{interrupts/sec})^{-1}$

The optimal performance setting for this register is very system and configuration specific.

CFI

The Extended Interrupt Throttle register should default to 0b upon initialization and reset. It loads in the value programmed by the software after software initializes the device.

The 82598 implements interrupt moderation to reduce the number of interrupts software processes. The moderation scheme is based on the EITR. 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 the proper MSI-X vector that supports the EICR bit has counted down to zero. The EITR counter is reloaded after it has reached zero with its initial value and the process repeats again. The interrupt flow should follow the following diagram:

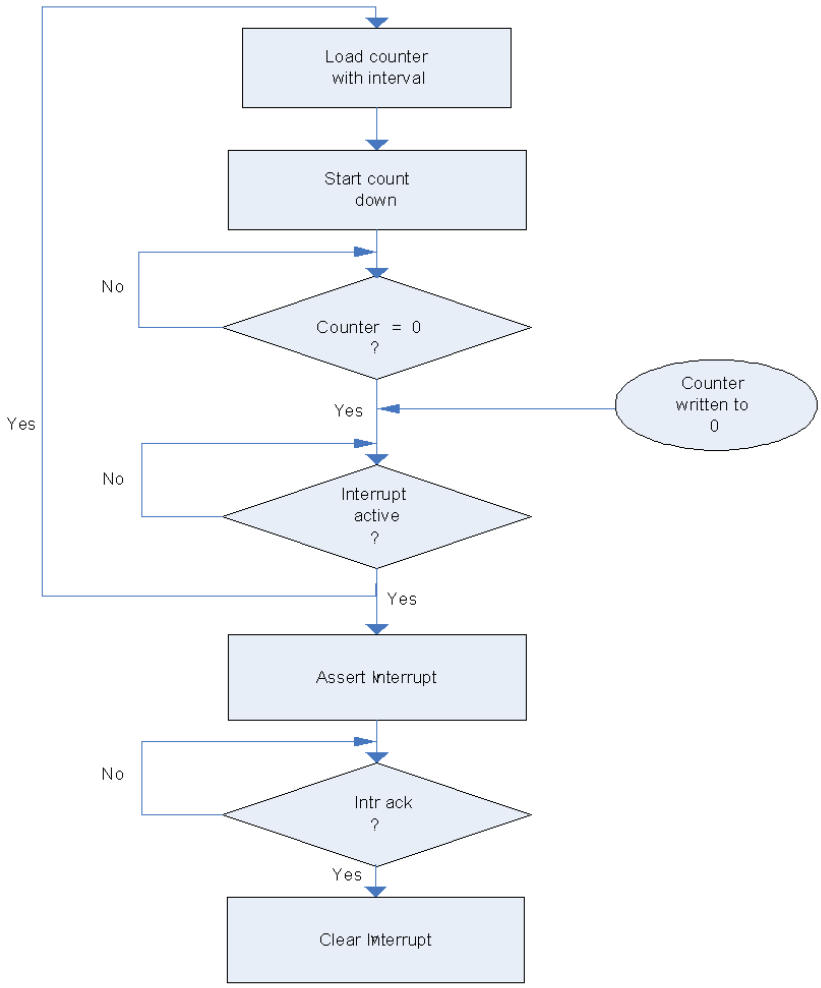
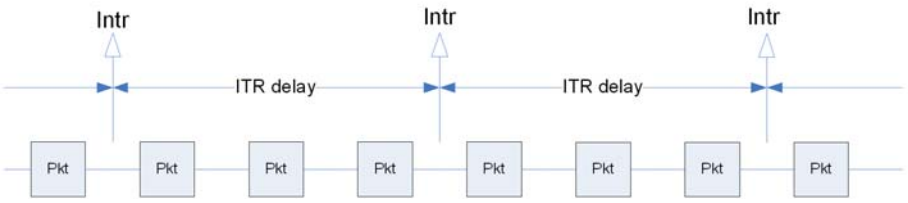


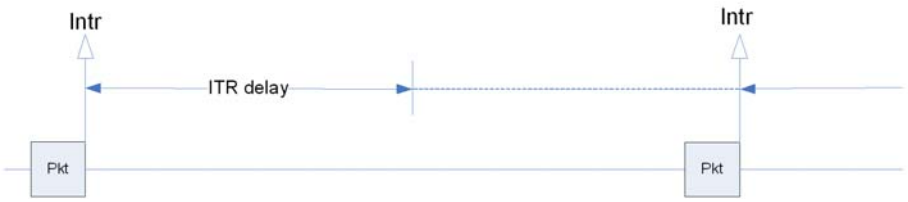
Figure 3-39. Interrupt Throttle Flow Diagram

For cases where the 82598 is connected to a small number of clients, it is desirable to initialize the interrupt as soon as possible with minimum latency. For these cases, when the EITR counter counts down to zero and no interrupt event has happened, then the EITR counter is not reset but stays at zero. Thus, the next interrupt event triggers an interrupt immediately. That scenario is illustrated as Case B.

Case A: Heavy load, interrupts moderated



Case B: Light load, interrupts immediately on packet receive



Note: To ensure the interrupts rate is properly controlled by software and is not affected by EICR reads, the EITR restarts counting for the next interrupt trigger right after the interrupt trigger and does not wait for the interrupt to be cleared as it used to wait in previous devices.

3.5.4.3 Clearing Interrupt Causes

The 82598 has three methods available for to clear EICR bits: Autoclear, clear-on-write, and clear-on-read.

Auto-Clear

In systems that support MSI-X, the interrupt vector enables the interrupt service routine to know the interrupt cause without reading the EICR. With interrupt moderation active, software loads 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 autoclear mode by setting the corresponding bits in the Extended Interrupt Auto-Clear (EIAC) register.

When auto-clear is enabled for a interrupt cause, the EICR bit is set when a cause event 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 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.

Write to Clear

The EICR register clears specific interrupt cause bits in the register after writing 1b to those bits. Any bit that was written with a 0b remains unchanged.

Read to Clear

All bits in the EICR register are cleared on a read to EICR If GPIE.OCD is not set. If set, only the other causes bits are cleared on read.

3.5.4.4 Dynamic Interrupt Moderation

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 when a packet is received by hardware and when it is indicated to the host operating system. This traffic can be identified by the TCP port value, in conjunction with control bits, size, and VLAN priority.

The 82598 implements eight entries, software programmable, table of TCP ports and eight registers with control bits filter and size threshold. In addition, a dedicated register enables setting of VLAN priority threshold. If a packet is received on one of these TCP ports, and the conditions set by the register fit to the packet, Hardware should interrupt immediately, overriding the interrupt moderation by the EITR counter.

A *Port Enabling* bit allows enabling or disabling of a specific port for this purpose.

3.5.4.4.1 Implementation

The logic of the dynamic interrupt moderation is as follows:

- There are eight port filters. Each filter checks the value of incoming packets TCP port, size and control bits, against values stored in filter's register. Each parameter can be bypassed (or wild carded). 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 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 port filters.

Note that EITR is reset to 0b following a dynamic interrupt.

Note: Packets that are dropped or have errors do not cause an immediate interrupt.

3.5.4.5 TCP Timer Interrupt

In order to implement TCP timers for I/OAT, software needs to take action periodically (every 10 ms). The software device driver must rely on software-based timers, whose granularity can change from platform to platform. This software timer generates a software NIC interrupt, which then enables the software device driver to perform timer functions as part of its usual DPC, avoiding cache thrash and enabling parallelization. The timer interval is system-specific.

The software device driver programs 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 software device driver, and it should be able to disable the timer interrupt if it is not needed.

3.5.4.5.1 Description

A stand-alone down-counter is implemented. An interrupt is issued each time the value of the counter is zero.

Software is responsible for setting the initial value for the timer in the *Duration* field. Kick-starting is done by writing 1b to the *KickStart* bit.

Following kick-starting, an internal counter is set to the value defined by the *Duration* field. Then the counter is decreased by one each ms. When the counter reaches zero, an interrupt is issued. The counter re-starts counting from its initial value if the *Loop* field is set.

3.5.4.6 MSI-X Interrupts

MSI-X defines a separate optional extension to basic MSI functionality. Compared to MSI, MSI-X supports a larger maximum number of vectors per function, 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 82598 interrupt causes into an interrupt vector that is conveyed by the 82598 as a posted-write PCIe transaction. Mapping of an interrupt cause into an MSI-X vector is determined by system software (device driver) through a translation table stored in the MSI-X allocation registers. Each entry of the allocation registers define the vector for a single interrupt cause. Table 3-77 lists which interrupt cause is represented by each entry in the MSI-X Allocation registers.

Table 3-77. Interrupt Cases for MSI-X

Interrupt	Entry[1]	Description
RxQ[63:0]	63:0	Receive Queues Associates an interrupt occurring in each of the Rx queues with a corresponding entry in the MSI-X Allocation registers.
TxQ[31:0]	95:64	Transmit Queues Associates an interrupt occurring in each of the Tx queues with a corresponding entry in the MSI-X Allocation registers.
TCP Timer	96	TCP Timer Associates an interrupt issued by the TCP timer with a corresponding entry in the MSI-X Allocation registers
Other causes	97	Other Causes Associates an interrupt issued by the other causes with a corresponding entry in the MSI-X Allocation registers

1. Entry in the MSI-X Allocation registers.

Each MSI-X interrupt vector has some attributes assigned to it, such as the address and data for its posted-write message.

3.5.5 802.1q VLAN Support

The 82598 provides several specific mechanisms to support 802.1q VLANs:

- Optional adding (for transmits) and ping strip (for receives) of IEEE 802.1q VLAN tags.
- Optional ability to filter packets belonging to certain 802.1q VLANs.

3.5.5.1 802.1q VLAN Packet Format

The following table compares an untagged 802.3 Ethernet packet with an 802.1q VLAN tagged packet:

Table 3-78. Comparing Packets

802.3 Packet	#Octets
DA	6
SA	6

802.1q VLAN Packet	#Octets
DA	6
SA	6

Type/Length	2
Data	46-1500
CRC	4

802.1q Tag	4
Type/Length	2
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.

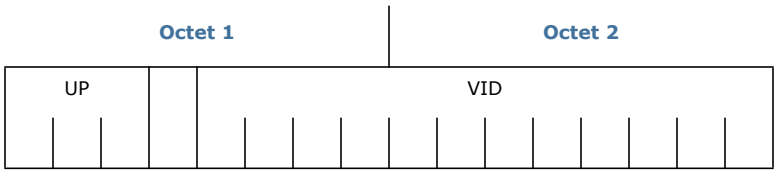
3.5.5.2 802.1q Tagged Frames

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 two octets. The first 16 bits of the tag header makes up the TPID. It contains the protocol type that 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 VLNCTRL.
- VLAN Identifier (VID)

The bit ordering is as follows:



3.5.5.3 Transmitting and Receiving 802.1q Packets

Since the 802.1q tag is only four bytes, adding and stripping of tags could be done completely in software. (In other words, for transmits, software inserts the tag into packet data before it builds the transmit descriptor list, and for receives, software strips the 4-byte tag from the packet data before delivering the packet to upper layer software) However, because adding and stripping of tags in software adds over-head for the host, the 82598 has additional capabilities to add and strip tags in hardware. See Section 3.5.5.3.1 and Section 3.5.5.3.2.

3.5.5.3.1 Adding 802.1q Tags on Transmits

Software might command the 82598 to insert an 802.1q VLAN tag on a per packet basis. If the *VLE* bit in the transmit descriptor is set to 1b, then the 82598 inserts a VLAN tag into the packet that it transmits over the wire. The *TPID* field of the 802.1q tag comes from the VET register, and the TCI of the 802.1q tag comes from the VLAN field of the legacy transmit descriptor or the *VLAN Tag* field of the advanced transmit descriptor. Refer to Table 3-65 for more information regarding hardware insertion of tags for transmits.

3.5.5.3.2 Stripping 802.1q Tags on Receives

Software might instruct the 82598 to strip 802.1q VLAN tags from received packets. If the VLNCTRL.VME bit is set to 1b, and the incoming packet is an 802.1q VLAN packet (it's Ethernet Type field matched the VET), then the 82598 strips the 4-byte VLAN tag from the packet and stores the TCI in the *VLAN Tag* field of the receive descriptor.

The 82598 also sets the *VP* bit in the receive descriptor to indicate that the packet had a VLAN tag that was stripped. If the VLNCTRL.VME bit is not set, the 802.1q packets can still be received if they pass the receive filter, but the VLAN tag is not stripped and the *VP* bit is not set. Refer to Table 3-79 for more information regarding receive packet filtering.

3.5.5.4 802.1q VLAN Packet Filtering

VLAN filtering is enabled by setting the VLNCTRL.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 for acceptance.

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 4096-bit vector is comprised of 128, 32-bit registers. Matching to this bit vector follows the same algorithm as for Multicast Address filtering. The VLAN Identifier (VID) field consists of 12 bits. The upper seven bits of this field are decoded to determine the 32-bit register in the VLAN Filter Table Array to address and the lower five bits determine which of the 32 bits in the register to evaluate for matching.

Two other bits in the VLNCTRL register, CFIEN and CFI, are also used in conjunction with 802.1q VLAN filtering operations. CFIEN enables the comparison of the value of the CFI bit in the 802.1q packet to the Receive Control register *CFI* bit as acceptance criteria for the packet.

Note: The *VFE* bit does not effect whether the VLAN tag is stripped. It only effects whether the VLAN packet passes the receive filter.

Table 3-79 lists reception actions per control bit settings.

Table 3-79. Packet Reception Decision Table

Is packet 802.1q?	VLNCTRL. VME	VLNCTRL. VFE	ACTION
No	X	X	Normal packet reception
Yes	0b	0b	Receive a VLAN packet if it passes the standard MAC address filters (only). Leave the packet as received in the data buffer. *VP* bit in receive descriptor is cleared.
Yes	0b	1b	Receive a VLAN packet if it passes the standard filters and the VLAN filter table. Leave the packet as received in the data buffer (the VLAN tag would not be stripped). *VP* bit in receive descriptor is cleared.
Yes	1b	0b	Receive a VLAN packet if it passes the standard filters (only). Strip off the VLAN information (four bytes) from the incoming packet and store in the descriptor. Sets *VP* bit in receive descriptor.
Yes	1b	1b	Receive a VLAN packet if it passes the standard filters and the VLAN filter table. Strip off the VLAN information (four bytes) from the incoming packet and store in the descriptor. Sets VP bit in receive descriptor.

Note: A packet is defined as a VLAN/802.1q packet if its *Type* field matches the VET.

3.5.6 DCA

3.5.6.1 Description

Direct Cache Access (DCA) is a method to improve network I/O performance by placing some posted inbound writes directly within CPU cache. DCA potentially eliminates cache misses due to inbound writes.

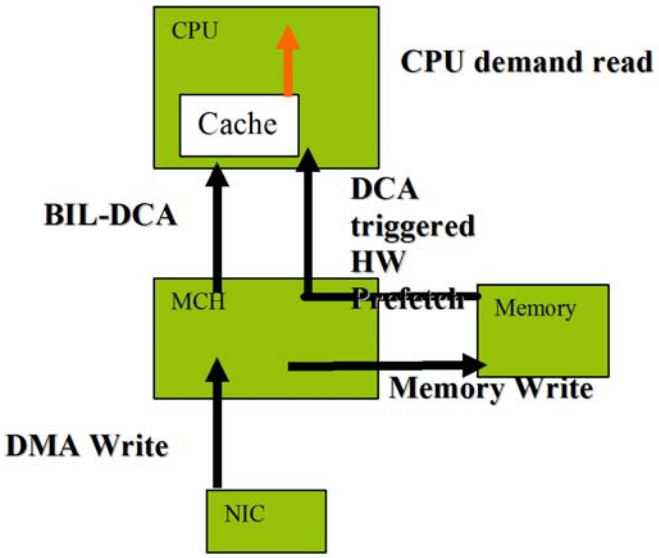


Figure 3-40. DCA Implementation on FSB System

As Figure 3-40 illustrates, 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 at power-on reset. A software device driver for the I/O device configures the I/O device for DCA and sets up the appropriate CPU ID and bus 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 to the CPU cache.

DCA implementation is controlled by separated registers (DCA_RXCTRL and DCA_TXCTRL) the assignment of receive queues to DCA_RXCTL is described in Section 3.5.2, the assignment of transmit queues to DCA_TXCTL is described in the following – DCA_TXCTRL0 is assigned to transmit queues 0 to 16. DCA_TXCTRL1 is assigned to transmit queues 1 to 17 … DCA_TXCTRL15 is assigned to transmit queues 15 to 31.

In addition, a DCA_ID register can be found for each port, in order to make visible the function, device, and bus numbers to the software device driver.

The DCA_RXCTRL and DCA_TXCTRL 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 has completed.

The DCA implemented in the 82598 makes use of the MWr method (as opposed to VDM method). This way, it is consistent with both generations for IOH/MCH.

However, in order to implement DCA, the 82598 has to be aware of the data movement engine version used (DME1/DME2). The software device driver initializes the 82598 to be aware of the bus configuration. A new register named DCA_CTRL is used in order to properly define the system configuration.

There are two modes for DCA implementation:

1. DME1: The DCA target ID is derived from CPU ID
2. DME2: The DCA target ID is derived from APIC ID.

The software device driver selects one of these modes through the DCA_mode register.

Both modes are described in the sections that follow.

3.5.6.2 PCIe Message Format for DCA (MWr Mode)

Figure 3-41 shows the format of the PCIe message for DCA.

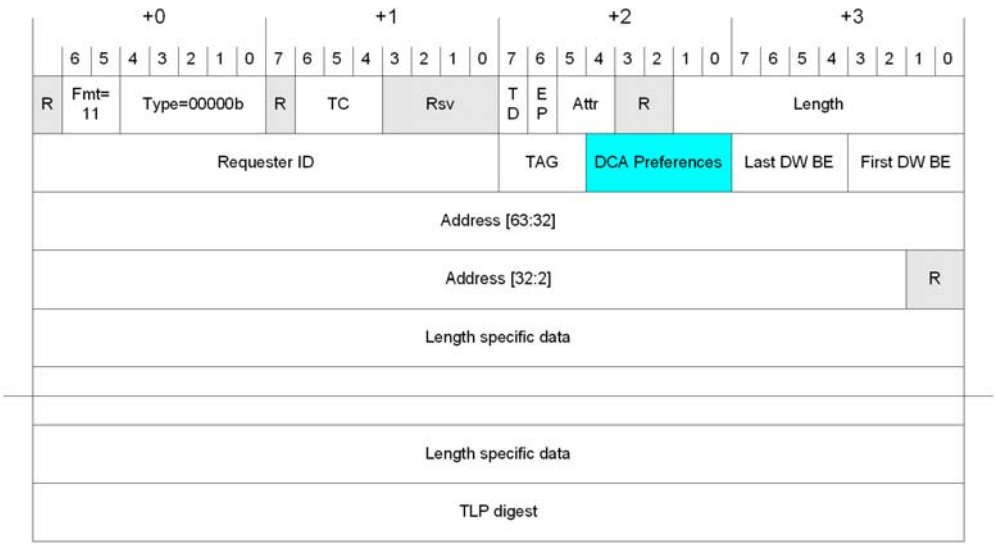


Figure 3-41. PCIe Message Format for DCA

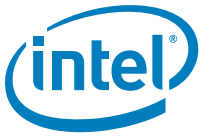
The DCA preferences field has the following formats.

For FSB chipset:

Bits	Name	Description
0	DCA indication	0b = DCA disabled 1b = DCA enabled
4:1	DCA Target ID	The DCA Target ID specifies the target cache for the data.

For CSI chipset:

Bits	Name	Description
4:0	DCA target ID	11111b: DCA is disabled Other: Target Core ID derived from APIC ID. The method for this is described in the DCA Platform Architecture Specification.



Note: All functions within the 82598 have to adhere to the tag encoding rules for DCA writes. Even if a given function is not capable of DCA, but other functions are capable of DCA, memory writes from the non-DCA function must set the tag field to 11111b.

3.5.7 LED's

The 82598 implements four output drivers intended for driving external LED circuits per port. Each of the four 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. In addition, the hardware-default configuration for all LED outputs can be specified via EEPROM fields thereby supporting LED displays configurable to a particular OEM preference.

Each of the four LED's can be configured to use one of a variety of sources for output indication.

The *IVRT* bits enable 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 200 ms, then off for 200 ms) while the LED source is asserted. Note that you must have link in order for the LEDs to blink. To ensure you have link, set the Force Link Up (FLU) bit when you want to blink the LEDs. When you want to stop blinking, reset the FLU bit to 0b. The blink control can be especially useful for ensuring that certain events, such as ACTIVITY indication, cause LED transitions, which are sufficiently visible by a human eye.

Note: The LINK/ACTIVITY source functions slightly different from the others when BLINK is enabled. The LED is off if there is no LINK, on if there is LINK and no ACTIVITY, and blinking if there is LINK and ACTIVITY.

§ §



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

4. Programming Interface

4.1 Address Regions

The 82598's address space is mapped into four regions along with the PCI Base Address registers. These regions are listed in Table 4-1.

Table 4-1. Address Regions

Addressable Content	Mapping Style	Region Size
Internal registers and memories	Direct memory mapped	128 kB
Flash (optional)	Direct memory-mapped	64-512 kB
Expansion ROM (optional)	Direct memory-mapped	64-512 kB
Internal registers and memories, Flash (optional)	I/O Window mapped	32 bytes
MSI-X (optional)	Direct Memory mapped	16 kB

Both the Flash and Expansion ROM Base Address registers map the same Flash memory. The internal registers, memories and Flash are be accessed though I/O space by doing a level of indirection.

4.2 Memory-Mapped Access

4.2.1 Memory-Mapped Access to Internal Registers and Memories

Internal registers and memories are be accessed as direct memory-mapped offsets from the base address register (BAR0 or BAR0/BAR1). See Section 4.4 for the appropriate offset for each internal register.

4.2.2 Memory-Mapped Accesses to Flash

External Flash is accessed using direct memory-mapped offsets from the Flash base address register (BAR1 or BAR2/BAR3). Flash is only accessible if enabled through the EEPROM Initialization Control Word, and if the Flash Base Address register contains a valid (non-zero) base memory address. For accesses, the offset from the Flash BAR corresponds to the offset into the Flash's actual physical memory space.

4.2.3 Memory-Mapped Access to Expansion ROM

External Flash can also 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 EEPROM Initialization Control Word and if the Expansion ROM Base Address register contains a valid (non-zero) base memory address.

4.3 I/O-Mapped Access

All internal registers, memories, and Flash can be accessed using I/O operations. I/O accesses are supported only if an I/O base address is allocated and mapped (BAR2 or BAR4), the BAR contains a valid value, and I/O address decoding is enabled in PCIe configuration.

When an I/O BAR is mapped, the I/O address range allocated opens a 32-byte window in the system I/0 address map. Within this window, two I/O addressable register are implemented: IOADDR and IODATA. The IOADDR register is used to specify a reference to an internal register, memory, or Flash; IODATA register is used as a window to the register, memory or Flash address specified by IOADDR.

Offset	Abbreviation	Name	RW	Size
0x00	IOADDR	Internal register, internal memory, or Flash location address. 0x00000-0x1FFFF – Internal registers/memories 0x20000-0x7FFFF – Undefined 0x80000-0xFFFFF – Flash	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 have read/write capability.	RW	4 bytes
0x08-0x1F	Reserved	Reserved	O	4 bytes

4.3.1 IOADDR (I/O Offset 0x00, RW)

IOADDR must always be written as a Dword access. Writes that are less than 32 bits are ignored. Reads of any size return a Dword; 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 32-bit, the source register of OUT 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 we recommended EAX be used.

Because only a particular range is addressable, the upper bits of this register are hard coded to zero. Bits 31 through 20 are not write-able and always read back as 0b.

On hardware reset (Internal Power On Reset or LAN_PWR_GOOD) or PCI Reset, this register value resets to 0x00000000. Once written, the value is retained until the next write or reset.

4.3.2 IODATA (I/O Offset 0x04, RW)

IODATA must always be written as a Dword access when the IOADDR register contains a value for internal registers and memories (such as 0x00000-0x1FFFC). Writes less than 32 bits are ignored.

The IODATA register can be written as a byte, word, or Dword access when the register contains a value for Flash (such as 0x80000-0xFFFFF). In this case, the IODATA value must be properly aligned to the data value. Additionally, the lower 2 bits of the IODATA PCI-X access must correspond to the byte, word, or Dword access. The following table lists the supported configurations.

Table 4-2. Supported IODATA Configurations

Access Type	IOADDR Register Bits [1:0]	Target IODATA Access BE[3:0]# bits in Data Phase
BYTE (8 bit)	00b	1110b
	01b	1101b
	10b	1011b
	11b	0111b
WORD (16 bit)	00b	1100b
	10b	0011b
DWORD (32 bit)	00b	0000b

Software might have to implement non-obvious code to access the Flash at a byte or word at a time. Example code that reads a Flash byte is shown:

```
char *IOADDR;
char *IODATA;
IOADDR = IOBASE + 0;
IODATA = IOBASE + 4;
*(IOADDR) = Flash_Byte_Address;
Read_Data = *(IODATA + (Flash_Byte_Address % 4));
```

Reads to IODATA of any size return a Dword; 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 OUT must be EAX (the only 32-bit register supported).

Writes and reads to IODATA when the IOADDR register value is in an undefined range (0x20000-0x7FFFC) should not be performed. Results cannot be determined.

Note: There are no special software timing requirements for accesses to IOADDR or IODATA. All accesses are immediate except when data is not readily available or acceptable. In this case, the 82598 delays results through normal bus methods (such as split transaction or transaction retry).

Because a register/memory/Flash read or write takes two I/O cycles, software must guarantee that the two I/O cycles occur as an atomic operation. Otherwise, results can be non-deterministic from a software viewpoint.

4.3.3 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 are discarded.

4.4 Device Registers

4.4.1 Terminology

Shorthand	Description
R/W	Read/Write. A register with this attribute can be read and written. If written since reset, the value read reflects the value written.
R/W S	Read/Write Status. A register with this attribute can be read and written. This bit represents status of some sort, so the value read may not reflect the value written.
RO	Read Only. If a register is read only, writes to this register have no effect.
WO	Write Only. Reading this register may not return a meaningful value.
R/WC	Read/Write Clear. A register bit with this attribute can be read and written. However, a Write of a 1b clears (sets to 0b) the corresponding bit and a write of a 0b has no effect.
R/Clr	Read Clear. A register bit with this attribute is cleared after read. Writes have no effect on the bit value.
R/W SC	Read/Write Self Clearing. When written to a 1b the bit causes an action to be initiated. Once the action is complete, the bit returns to 0b.
RO/LH	Read Only, Latch High. The bit records an event or the occurrence of a condition to be recorded. When the event occurs the bit is set to 1b. After the bit is read, it returns to 0b unless the event is still occurring.
RO/LL	Read Only, Latch Low. The bit records an event. When the event occurs the bit is set to 0b. After the bit is read, it reflects the current status.
RW0	Ignore Read, Write Zero. The bit is a reserved bit. Any values read should be ignored. When writing to this bit always write a 0b.
RWP	Ignore Read, Write Preserving. This bit is a reserved bit. Any values read should be ignored. However, they must be saved. When writing the register the value read out must be written back. (There are currently no bits that have this definition.)

4.4.2 Register List

The 82598's non-PCIe configuration registers are listed in Table 4-3. These registers are ordered by group and are not necessarily listed in the order that they appear in address space.

All registers should be accessed as a 32-bit width on reads with an appropriate software mask. Software read/modify/write mechanism should be invoked for partial writes.

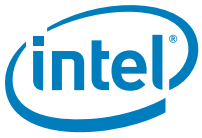
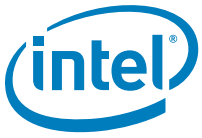


Table 4-3. Register List

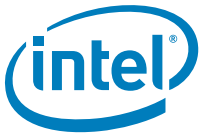
Category	Offset	Abbreviation	Register Name	RW	Page
General	0x00000 - 0x00004	CTRL	Device Control	RW	304
General	0x00008	STATUS	Device Status	RO	305
General	0x00018	CTRL_EXT	Extended Device Control	RW	305
General	0x0020	ESDP	Extended SDP Control	RW	306
General	0x0028	EODSDP	Extended OD SDP Control	RW	307
General	0x00200	LEDCTL	LED Control	RW	308
General	0x0004C	TCPTimer	TCP Timer	RW	310
NVM	0x10010	EEC	EEPROM/Flash Control	RW	310
NVM	0x10014	EERD	EEPROM Read	RW	313
NVM	0x1001C	FLA	Flash Access	RW	313
NVM	0x10110	EEMNGCTL	Manageability EEPROM Control	RW	314
NVM	0x10114	EEMNGDATA	Manageability EEPROM Read/Write Data	RW	315
NVM	0x10118	FLMNGCTL	Manageability Flash Control	RW	315
NVM	0x1011C	FLMNGDATA	Manageability Flash Read Data	RW	315
NVM	0x10120	FLMNGCNT	Manageability Flash Read Counter	RW	316
NVM	0x1013C	FLOP	Flash Opcode	RW	316
NVM	0x10200	GRC	General Receive Control	RW	317
Interrupt	0x00800	EICR	Extended Interrupt Cause Read	R/C	317
Interrupt	0x00808	EICS	Extended Interrupt Cause Set	WO	319
Interrupt	0x00880	EIMS	Extended Interrupt Mask Set/Read	RWS	319
Interrupt	0x00888	EIMC	Extended Interrupt Mask Clear	WO	320
Interrupt	0x00810	EIAC	Extended Interrupt Auto Clear	RW	321
Interrupt	0x00890	EIAM	Extended Interrupt Auto Mask Enable	RW	322
Interrupt	0x00820 - 0x0086C	EITR	Extended Interrupt Throttling Rate 0 - 19	RW	322
Interrupt	0x00900- 0x00960	IVAR	Interrupt Vector Allocation Registers	RW	323
Interrupt	BAR3/ 0x0000 - 0x013C	MSIXT[19:0]	MSI-X Table	RW	324
Interrupt	BAR3/ 0x2000	MSIXPBA	MSI-X Pending Bit Array	RO	324

Interrupt	0x11068	PBACL	MSI-X PBA Clear	RW	324
Interrupt	0x00898	GPIE	General Purpose Interrupt Enable	RW	325
Flow Control	0x03008	PFCTOP	Priority Flow Control Type Opcode	RW	326
Flow Control	0x03200 – 0x0320C	FCTTV[0-3]	Flow Control Transmit Timer Value	RW	326
Flow Control	0x03220 + n*0x8	FCRTL[0-7]	Flow Control Receive Threshold Low	RW	326
Flow Control	0x03260 + n*0x8	FCRTH[0-7]	Flow Control Receive Threshold High	RW	327
Flow Control	0x032A0	FCRTV	Flow Control Refresh Threshold Value	RW	328
Flow Control	0x0CE00	TFCS	Transmit Flow Control Status	RO	328
Receive DMA	0x01000+ n*0x40	RDBAL[0-63]	Rx Descriptor Base Low	RW	329
Receive DMA	0x01004+ n*0x40	RDBAH[0-63]	Rx Descriptor Base High	RW	329
Receive DMA	0x01008+ n*0x40	RDLEN[0-63]	Rx Descriptor Length	RW	329
Receive DMA	0x01010+ n*0x40	RDH[0-63]	Rx Descriptor Head	RO	329
Receive DMA	0x01018+ n*0x40	RDT[0-63]	Rx Descriptor Tail	RW	330
Receive DMA	0x01028+ n*0x40	RXDCTL[0-63]	Receive Descriptor Control	RW	330
Receive DMA	0x02100 – 0x0213C	SRRCTL[0-15]	Split Receive Control	RW	332
Receive DMA	0x02200 – 0x0223C	DCA_RXCTRL [0-15]	Rx DCA Control	RW	332
Receive DMA	0x02F00	RDRXCTL	Receive DMA Control	RW	333
Receive DMA	0x03C00 – 0x03C1C	RXPBSIZE	Receive Packet Buffer Size	RW	334
Receive DMA	0x03000	RXCTRL	Receive Control	RW	334
Receive DMA	0x03D04 – 0x03D08	DROPEN	Drop Enable Control	RW	335
Receive	0x05000	RXCSUM	Receive Checksum Control	RW	335
Receive	0x05008	RFCTL	Receive Filter Control	RW	336
Receive	0x05200- 0x053FC	MTA[127:0]	Multicast Table Array (n)	RW	336
Receive	0x05400	RAL(0)	Receive Address Low (0)	RW	337
Receive	0x05404	RAH(0)	Receive Address High (0)	RW	337



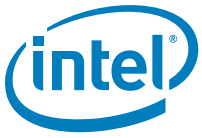
	…	…	…	…	
Receive	0x05478	RAL(15)	Receive Address Low (15)	RW	337
Receive	0x0547C	RAH(15)	Receive Address High (15)	RW	337
Receive	0x05480 – 0x054BC	PSRTYPE	Packet Split Receive Type	RW	338
Receive	0x0A000-0x0A9FC	VFTA	VLAN Filter Table Array	RW	339
Receive	0x05080	FCTRL	Filter Control	RW	341
Receive	0x05088	VLNCTRL	VLAN Control	RW	343
Receive	0x05090	MCSTCTRL	Multicast Control	RW	343
Receive	0x05818	MRQC	Multiple Receive Queues Command	RW	343
Receive	0x0581C	VMD_CTL	VMDq Control	RW	344
Receive	0x05A80 - 0x05A9C	IMIR	Immediate Interrupt Rx [7:0]	RW	344
Receive	0x05AA0 - 0x05ABC	IMIREXT	Immediate Interrupt Rx Extended[0-7]	RW	345
Receive	0x05AC0	IMIRVP	Immediate Interrupt Rx VLAN Priority	RW	345
Receive	0x05C00-0x05C3F	RETA	Redirection Table	RW	346
Receive	0x05C80-0x05CAF	RSSRK	RSS Random Key Register	RW	347
Transmit	0x06000+ n*0x40	TDBAL[0-31]	Tx Descriptor Base Low	RW	347
Transmit	0x06004+ n*0x40	TDBAH[0-31]	Tx Descriptor Base High	RW	348
Transmit	0x06008+ n*0x40	TDLEN[0-31]	Tx Descriptor Length	RW	348
Transmit	0x06010+ n*0x40	TDH[0-31]	Tx Descriptor Head	RO	348
Transmit	0x06018+ n*0x40	TDT[0-31]	Tx Descriptor Tail	RW	348
Transmit	0x06028+ n*0x40	TXDCTL[0-31]	Transmit Descriptor Control	RW	349
Transmit	0x06038+ n*0x40	TDWBAL[0-31]	Transmit Descriptor Write Back Address Low	RW	350
Transmit	0x0603C+n* 0x40	TDWBAH[0-31]	Transmit Descriptor Write Back Address High	RW	350
Transmit	0x07E00	DTXCTL	DMA Tx Control	RW	350
Transmit	0x07200 – 0x0723C	DCA_TXCTRL[0-15]	Tx DCA CTRL Register	RW	390

Transmit	0x0CB00	TIPG	Transmit IPG Control	RW	351
Transmit	0x0CC00 – 0x0CC1C	TXPBSIZE	Transmit Packet Buffer Size	RW	352
Transmit	0x0CD10	MNGTXMAP	Manageability Transmit TC Mapping	RW	352
Wake	0x05800	WUC	Wake Up Control	RW	352
Wake	0x05808	WUFC	Wake Up Filter Control	RW	353
Wake	0x05810	WUS	Wake Up Status	RO	353
Wake	0x05838	IPAV	IP Address Valid	RW	354
Wake	0x05840-0x05858	IP4AT	IPv4 Address Table	RW	355
Wake	0x05880-0x0588F	IP6AT	IPv6 Address Table	RW	355
Wake	0x05900	WUPL	Wake Up Packet Length	RW	356
Wake	0x05A00-0x05A7C	WUPM	Wake Up Packet Memory	R	356
Wake	0x09000-0x093FC	FHFT	Flexible Host Filter Table	RW	356
Statistics	0x04000	CRCERRS	CRC Error Count	RO	358
Statistics	0x04004	ILLERRC	Illegal Byte Error Count	RO	358
Statistics	0x04008	ERRBC	Error Byte Count	RO	358
Statistics	0x04010	MSPDC	MAC Short Packet Discard Count	RO	358
Statistics	0x03FA0 – 0x03FBC	MPC	Missed Packets Count[0-7]	RO	359
Statistics	0x04034	MLFC	MAC Local Fault Count	RO	359
Statistics	0x04038	MRFC	MAC Remote Fault Count	RO	359
Statistics	0x04040	RLEC	Receive Length Error Count	RO	359
Statistics	0x03F60	LXONTXC	Link XON Transmitted Count	RO	359
Statistics	0x0CF60	LXONRXC	Link XON Received Count	RO	360
Statistics	0x03F68	LXOFFTXC	Link XOFF Transmitted Count	RO	360
Statistics	0x0CF68	LXOFFRXC	Link XOFF Received Count	RO	360
Statistics	0x03F00 – 0x03F1C	PXONTXC	Priority XON Transmitted Count[0-7]	RO	360
Statistics	0x0CF00 – 0x0CF1C	PXONRXC	Priority XON Received Count[0-7]	RO	360
Statistics	0x03F20 – 0x03F3C	PXOFFTXC	Priority XOFF Transmitted Count[0-7]	RO	361



Statistics	0x0CF20 – 0x0CF3C	PXOFFRXC	Priority XOFF Received Count[0-7]	RO	361
Statistics	0x0405C	PRC64	Packets Received (64 Bytes) Count	RO	361
Statistics	0x04060	PRC127	Packets Received (65-127 Bytes) Count	RO	361
Statistics	0x04064	PRC255	Packets Received (128-255 Bytes) Count	RO	362
Statistics	0x04068	PRC511	Packets Received (256-511 Bytes) Count	RO	362
Statistics	0x0406C	PRC1023	Packets Received (512-1023 Bytes) Count	RO	362
Statistics	0x04070	PRC1522	Packets Received (1024-1522 Bytes)	RO	362
Statistics	0x04074	GPRC	Good Packets Received Count	RO	363
Statistics	0x04078	BPRC	Broadcast Packets Received Count	RO	363
Statistics	0x0407C	MPRC	Multicast Packets Received Count	RO	363
Statistics	0x04080	GPTC	Good Packets Transmitted Count	RO	363
Statistics	0x0408C	GORC	Good Octets Received Count	RO	364
Statistics	0x04094	GOTC	Good Octets Transmitted Count	RO	364
Statistics	0x03FC0 – 0x03FDC	RNBC	Receive No Buffers Count[0-7]	RO	364
Statistics	0x040A4	RUC	Receive Undersize Count	RO	365
Statistics	0x040A8	RFC	Receive Fragment Count	RO	365
Statistics	0x040AC	ROC	Receive Oversize Count	RO	365
Statistics	0x040B0	RJC	Receive Jabber Count	RO	365
Statistics	0x040B4	MNGPRC	Management Packets Receive Count	RO	366
Statistics	0x040B8	MNGPDC	Management Packets Dropped Count	RO	366
Statistics	0x0CF90	MNGPTC	Management Packets Transmitted Count	RO	366
Statistics	0x040C4	TOR	Total Octets Received (High)	RO	366
Statistics	0x040D0	TPR	Total Packets Received	RO	367
Statistics	0x040D4	TPT	Total Packets transmitted	RO	367
Statistics	0x040D8	PTC64	Packets Transmitted (64 Bytes) Count	RO	367
Statistics	0x040DC	PTC127	Packets Transmitted (65-127 Bytes) Count	RO	367
Statistics	0x040E0	PTC255	Packets Transmitted (128-256 Bytes) Count	RO	368
Statistics	0x040E4	PTC511	Packets Transmitted (256-511 Bytes) Count	RO	368

Statistics	0x040E8	PTC1023	Packets Transmitted (512-1023 Bytes) Count	RO	368
Statistics	0x040EC	PTC1522	Packets Transmitted (1024-1522 Bytes) Count	RO	368
Statistics	0x040F0	MPTC	Multicast Packets Transmitted Count	RO	369
Statistics	0x040F4	BPTC	Broadcast Packets Transmitted Count	RO	369
Statistics	0x04120	XEC	XSUM Error Count	RO	369
Statistics	0x02300 + n*0x4	RQSMR	Receive Queue Statistics Mapping [15:0]	RW	369
Statistics	0x07300 + n*0x4	TQSMR	Transmit Queue Statistics Mapping [7:0]	RW	370
Statistics	0x01030+ n*0x40	QPRC	Queue Packets Received Count [15:0]	RO	371
Statistics	0x06030 + n*0x40	QPTC	Queue Packets Transmitted Count [15:0]	RO	371
Statistics	0x01034+ n*0x40	QBRC	Queue Bytes Received Count [15:0]	RO	371
Statistics	0x06034 + n*0x40	QBTC	Queue Bytes Transmitted Count [15:0]	RO	372
Management Filters	0x05010 – 0x0502C	MAVTV[7:0]	VLAN TAG Value [7:0]	RW	372
Management Filters	0x05030 - 0x0504C	MFUTP[7:0]	Management Flex UDP/TCP Ports	RW	372
Management Filters	0x05820	MANC	Management Control	RW	372
Management Filters	0x05824	MFVAL	Manageability Filters Valid	RW	373
Management Filters	0x05860	MANC2H	Management Control To Host	RW	374
Management Filters	0x05890 - 0x058AC	MDEF[7:0]	Manageability Filters Valid	RW	374
Management Filters	0x058B0 - 0x058EC	MIPAF	Manageability IP Address Filter	RW	376
Management Filters	0x05910	MMAL_0	Manageability MAC Address Low 0	RW	379
Management Filters	0x05914	MMAH_0	Manageability MAC Address High 0	RW	379
Management Filters	0x05928	MMAL_3	Manageability MAC Address Low 3	RW	379
Management Filters	0x0592C	MMAH_3	Manageability MAC Address High 3	RW	379
Management Filters	0x09400- 0x097FC	FTFT	Flexible TCO Filter Table	RW	379



PCIe	0x11000	GCR	PCIe* Control	RW	381
PCIe	0x11004	GTV	PCIe* Timer Value	RW	383
PCIe	0x11008	FUNCTAG	Function Tag	RW	383
PCIe	0x1100C	GLT	PCIe* Latency Timer	RW	384
PCIe	0x10150	FACTPS	Function Active and Power State	RO	384
PCIe	0x11040	PCIEANACTL	PCIe* Analog Configuration	RW	385
PCIe	0x10140	SWSM	Software Semaphore	RW	385
PCIe	0x10148	FWSM	Firmware Semaphore	RW	386
PCIe	0x10160	GSSR	General Software Semaphore Register	RW	387
PCIe	0x11064	MREVID	Mirrored Revision ID	RO	390
PCIe	0x11070	DCA_ID	DCA Requester ID Information	RO	390
PCIe	0x11074	DCA_CTRL	DCA Control	RW	390
MAC	0x04200	PCS1GCFIG	PCS_1G Global Configuration 1	RW	391
MAC	0x04208	PCS1GLCTL	PCS_1G Link Control	RW	391
MAC	0x0420C	PCS1GLSTA	PCS_1G Link Status	RO	392
MAC	0x04218	PCS1GANA	PCS_1G Auto Negotiation Advanced	RW	393
MAC	0x0421C	PCS1GANLP	PCS_1G AN LP Ability	RO	393
MAC	0x04220	PCS1GANNP	PCS_1G AN Next Page Transmit	RW	394
MAC	0x04224	PCS1GANLPNP	PCS_1G AN LP's Next Page	RO	395
MAC	0x04240	HLREG0	Flow Control 0	RW	396
MAC	0x04244	HLREG1	Flow Control Status 1	RO	397
MAC	0x04248	PAP	Pause and Pace	RW	398
MAC	0x0424C	MACA	MDI Auto Scan Command and Address	RW	399
MAC	0x04250	APAE	Auto-Scan PHY Address Enable	RW	399
MAC	0x04254	ARD	Auto-Scan Read Data	RW	399
MAC	0x04258	AIS	Auto-Scan Interrupt Status	RW	400
MAC	0x0425C	MSCA	MDI Signal Command and Address	RW	400
MAC	0x04260	MSRWD	MDI Single Read and Write Data	RW	401
MAC	0x04264	MLADD	Low MAC Address	RW	401
MAC	0x04268	MHADD	MAC Address High and Maximum Frame Size	RW	401
MAC	0x04288	PCSS1	XGXS Status 1	RO	401

MAC	0x0428C	PCSS2	XGXS Status 2	RO	402
MAC	0x04290	XPCSS	10GBase-X PCS Status	RO	402
MAC	0x04298	SERDESC	SerDes Interface Control	RW	404
MAC	0x0429C	MACS	FIFO Status/Control Report	RW	405
MAC	0x042A0	AUTOC	Auto-Detect Control/Status	RW	405
MAC	0x042A4	LINKS	Link Status	RO	407
MAC	0x042A8	AUTOC2	Auto Negotiation Control 2	RW	408
MAC	0x042AC	AUTOC3	Auto Negotiation Control 3	RW	409
MAC	0x042B0	ANLP1	Auto Negotiation Link Partner Control Word 1	RO	409
MAC	0x042B4	ANLP2	Auto Negotiation Link Partner Control Word 2	RO	409
MAC	0x042D0	MMNGC	MAC Manageability Control	Host-RO/ MNG-RW	410
MAC	0x042D4	ANLPNP1	Auto Negotiation Link Partner Next Page 1	RO	410
MAC	0x042D8	ANLPNP2	Auto Negotiation Link Partner Next Page 2	RO	411
MAC	0x04800	ATLASCTL	Core Analog Configuration	RW	411

4.4.3 Register Descriptions

4.4.3.1 General Control Registers

4.4.3.1.1 Device Control Register – CTRL (0x00000/0x00004, RW)

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0b	Reserved Write as 0b for future compatibility.
PCIe Master Disable	2	0b	When set, the 82598 blocks new master requests, including manageability requests, by using this function. Once no master requests are pending by using this function, the *GIO Master Enable Status* bit is set.
LRST	3	1b	Link Reset This bit performs a reset of the MAC, PCS, and auto negotiation functions and the entire the 82598 10 GbE controller (software reset) resulting in a state nearly approximating the state following a power-up reset or internal PCIe reset, except for the system PCI configuration. Normally 0b, writing 1b initiates the reset. This bit is self-clearing. Also referred to as MAC reset.
Reserved	25:4	0b	Reserved

RST	26	0b	Device Reset This bit performs a reset of the 82598, resulting in a state nearly approximating the state following a power-up reset or internal PCIe reset, except for the system PCI configuration. Normally 0b, writing 1b initiates the reset. This bit is self-clearing. Also referred to as a software reset or global reset. **Note:** This bit does not reset the MAC, PCS, or auto negotiation functions.
Reserved	31:27	0x0	Reserved

LRST and RST are used to globally reset the 82598 10 GbE controller. This register is provided primarily as a software mechanism to recover from an indeterminate or suspected hung hardware state. Most registers (receive, transmit, interrupt, statistics, etc.) and state machines are set to their power-on reset values, approximating the state following a power-on or PCI reset. However, PCIe configuration registers are not reset; this leaves the 82598 mapped into system memory space and accessible by a software device driver.

To ensure that a global device reset has fully completed and that the 82598 responds to subsequent accesses, programmers must wait approximately 1 ms (after setting) before checking if the bit has cleared or to access (read or write) device registers.

4.4.3.1.2 Device Status Register – STATUS (0x00008; R)

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0b	Reserved Read as 0b.
LAN ID	3:2	0b	LAN ID. Provides software a mechanism to determine the device LAN identifier for this MAC. Read as: [0,0] LAN 0, [0,1] LAN 1.
Reserved	18:4	0b	Reserved
PCIe Master Enable Status	19	1b	This is a status bit of the appropriate CTRL.GIO Master Disable bit. 1b = Associated LAN function can issue master requests. 0b = Associated LAN function does not issue any master request and all previously issued requests are complete.
Reserved	31:20	0b	Reserved Reads as 0b.

4.4.3.1.3 Extended Device Control Register – CTRL_EXT (0x00018; RW)

Field	Bit(s)	Initial Value	Description
Reserved	15:0	0b	Reserved
NS_DIS	16	0b	No Snoop Disable When set to 1b, the 82598 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 Disable. When set to 1b, the 82598 does not request any relaxed ordering transactions in PCIe mode regardless of the state of bit 1 in the PCIe command register. When this bit is clear and bit 1 of the PCIe command register is set, the 82598 requests relaxed ordering transactions as described in Section 4.4.3.5.8 and Section 4.4.3.12.1 (per queue and per flow).
Reserved	27:18	0b	Reserved
DRV_LOAD	28	0b	Driver loaded and the corresponding network interface is enabled. This bit should be set by the software device driver after it was loaded and cleared when it unloads or at PCIe soft reset. The BMC loads this bit as an indication that the software device driver successfully loaded to it.
Reserved	31:29	0b	Reserved Reads as 0b.

4.4.3.1.4 Extended SDP Control – ESDP (0x00020, RW)

Field	Bit(s)	Initial Value	Description
SDP0_DATA	0	0b[1]	SDP0 Data Value Used to read (write) a value of the 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. If SDP0 is configured as an input, all reads return the current value of the pin.
SDP1_DATA	1	0b[1]	SDP1 Data Value Used to read (write) a value of the 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. If SDP1 is configured as an input, all reads return the current value of the pin.
SDP2_DATA	2	0b[1]	SDP2 Data Value Used to read (write) a value of software-controlled I/O pin SDP2. If SDP2 is configured as an output (SDP2_IODIR = 1b), this bit controls the value driven on the pin. If SDP2 is configured as an input, all reads return the current value of the pin.
SDP3_DATA	3	0b[1]	SDP3 Data Value Used to read (write) a value of the software-controlled I/O pin SDP3. If SDP3 is configured as an output (SDP3_IODIR = 1b), this bit controls the value driven on the pin. If SDP3 is configured as an input, all reads return the current value of the pin.
SDP4_DATA	4	0b	SDP4 Data Value Used to read (write) a value of the software-controlled I/O pin SDP4. If SDP4 is configured as an output (SDP4_IODIR = 1b), this bit controls the value driven on the pin. If SDP4 is configured as an input, all reads return the current value of the pin.
SDP5_DATA	5	0b	SDP5 Data Value Used to read (write) a value of the software-controlled I/O pin SDP5. If SDP5 is configured as an output (SDP5_IODIR = 1b), this bit controls the value driven on the pin. If SDP5 is configured as an input, all reads return the current value of the pin.
Reserved	7:6	0x0	Reserved
SDP0_IODIR	8	0b[1]	SDP0 Pin Directionality Controls whether or not software-controlled pin SDP0 is configured as an input or output (0b = input, 1b = output). This bit is not affected by software or system reset, only by initial power-on or direct software writes.

SDP1_IODIR	9	0b[1]	SDP1 Pin Directionality Controls whether or not software-controlled pin SDP1 is configured as an input or output (0b = input, 1b = output). This bit is not affected by software or system reset, only by initial power-on or direct software writes.
SDP2_IODIR	10	0b[1]	SDP2 Pin Directionality Controls whether or not software-controlled pin SDP2 is configured as an input or output (0b = input, 1b = output). This bit is not affected by software or system reset, only by initial power-on or direct software writes.
SDP3_IODIR	11	0b[1]	SDP3 Pin Directionality Controls whether or not software-controlled pin SDP3 is configured as an input or output (0b = input, 1b = output). This bit is not affected by software or system reset, only by initial power-on or direct software writes.
Field	**Bit(s)**	**Initial Value**	**Description**
SDP4_IODIR	12	0b	SDP4 Pin Directionality Controls whether or not software-controlled pin SDP4 is configured as an input or output (0b = input, 1b = output). This bit is not affected by software or system reset, only by initial power-on or direct software writes.
SDP5_IODIR	13	0b	SDP5 Pin Directionality Controls whether or not software-controlled pin SDP5 is configured as an input or output (0b = input, 1b = output). This bit is not affected by software or system reset, only by initial power-on or direct software writes.
Reserved	31:14	0x0	Reserved

1. Initial value can be configured using the EEPROM.

4.4.3.1.5 Extended OD SDP Control – EODSDP (0x00028; RW)

Field	Bit(s)	Initial Value	Description
SDP6_DATA_IN	0	0b	SDP6 Data In Value Provides the value of SDP6 (input from external PAD).
SDP6_DATA_OUT	1	0b	SDP6 Data Out Value Used to drive the value of SDP6 (output to PAD).
SDP7_DATA_IN	2	0b	SDP7 Data In Value Provides the value of SDP7 (input from external PAD).
SDP7_DATA_OUT	3	0b	SDP7 Data Out Value Used to drive the value of SDP7 (output to PAD).
Reserved	31:4	0x0	Reserved

4.4.3.1.6 LED Control – LEDCTL (0x00200; RW)

Field	Bit(s)	Initial Value	Description
LED0_MODE	3:0	0x0[1]	LED0 Mode This field specifies the control source for the LED0 output. An initial value of 0000b selects the LINK_UP indication.
Reserved	4	0b[1]	Reserved
GLOBAL_BLINK_MODE	5	0b[1]	Global Blink Mode This field specifies the blink mode of all LEDs. 0b = Blink at 200 ms on and 200 ms off. 1b = Blink at 83 ms on and 83 ms off.
LED0_IVRT	6	0b[1]	LED0 Invert This field specifies the polarity/inversion of the LED source prior to output or blink control. 0b = Do not invert LED source. 1b = Invert LED source.
LED0_BLINK	7	0b[1]	LED0 Blink This field specifies whether or not to apply blink logic to the (inverted) LED control source prior to the LED output. 0b = Do not blink LED output. 1b = Blink LED output.
LED1_MODE	11:8	0001b[1]	LED1 Mode This field specifies the control source for the LED1 output. An initial value of 0001b selects the 10 Gb/s link indication.
Reserved	13:12	0b[1]	Reserved
LED1_IVRT	14	0b[1]	LED1 Invert This field specifies the polarity/inversion of the LED source prior to output or blink control. 0b = Do not invert LED source. 1b = Invert LED source.
LED1_BLINK	15	1b[1]	LED1 Blink This field specifies whether or not to apply blink logic to the (inverted) LED control source prior to the LED output. 0b = Do not blink LED output. 1b = Blink LED output.
LED2_MODE	19:16	0100b[1]	LED2 Mode. This field specifies the control source for the LED2 output. An initial value of 0100b selects LINK/ACTIVITY indication.
Reserved	21:20	00b[1]	Reserved
LED2_IVRT	22	0[1]	LED2 Invert This field specifies the polarity/inversion of the LED source prior to output or blink control. 0b = Do not invert LED source. 1b = Invert LED source.
LED2_BLINK	23	0[1]	LED2 Blink This field specifies whether or not to apply blink logic to the (inverted) LED control source prior to the LED output. 0b = Do not blink LED output. 1b = Blink LED output.

Field	Bit(s)	Initial Value	Description
LED3_MODE	27:24	0101b[1]	LED3 Mode This field specifies the control source for the LED3 output. An initial value of 0101b selects the 1 Gb/s link indication.
Reserved	29:28	0b[1]	Reserved
LED3_IVRT	30	0b[1]	LED3 Invert This field specifies the polarity/inversion of the LED source prior to output or blink control. 0b = Do not invert LED source. 1b = Invert LED source.
LED3_BLINK	31	0b[1]	LED3 Blink This field specifies whether or not to apply blink logic to the (inverted) LED control source prior to the LED output. 0b = Do not blink LED output. 1b = Blink LED output.

1. These bits are read from the EEPROM.

The following mapping is used to specify the LED control source (MODE) for each LED output.

MODE	Selected Mode	Source Indication
0000b	LINK_UP	Asserted when any speed link is established and maintained.
0001b	LINK_10G	Asserted when a 10 Gb/s link is established and maintained.
0010b	MAC_ACTIVITY	Asserted when link is established and packets are being transmitted or received.
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 there is no transmit or receive activity.
0101b	LINK_1G	Asserted when a 1 Gb/s link is established and maintained.
0110b:1101b	Reserved	Reserved
1110b	LED_ON	Always asserted.
1111b	LED_OFF	Always de-asserted.

Note: The dynamic LED modes (FILTER_ACTIVITY, LINK/ACTIVITY, and MAC_ACTIVITY) should be used with LED Blink mode enabled.

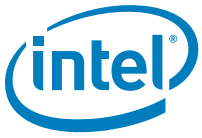
4.4.3.1.7 TCP_Timer - TCPTIMER (0x0004C, RW)

Field	Bit(s)	Initial Value	Description
Duration	7:0	0x0	Duration Duration of the TCP interrupt interval in ms.
KickStart	8	0b	Counter Kick-Start Writing 1b to this bit kick-starts the counter down-count from the initial value defined in *Duration* field. Writing 0b has no effect (WS).
TCPCountEn	9	0b	TCP Count Enable When 1b, TCP timer counting is enabled. When 0b, it is disabled. Upon enabling, TCP counter counts from its internal state. If the internal state is equal to zero, down-count does not restart until *KickStart* is activated. If the internal state is not 0b, down-count continues from the internal state. This enables a pause of the counting for debug purpose.
TCPCountFinish	10	0b	TCP Count Finish This bit enables software to trigger a TCP timer interrupt, regardless of the internal state. Writing 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 0b to this bit has no effect (WS).
Loop	11	0b	TCP Loop When 1b, TCP counter reloads duration each time it reaches zero and goes on down-counting from this point without kick-starting. When 0b, TCP counter stops at a zero value and does not re-start until *KickStart* is activated.
Reserved	31:12	0x0	Reserved

4.4.3.2 EEPROM/Flash Registers

4.4.3.2.1 EEPROM/Flash Control Register – EEC (0x10010; RW)

Field	Bit(s)	Initial Value	Description
EE_SK	0	0b	Clock input to the EEPROM When EE_GNT is set to 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	0b	Chip select input to the EEPROM When EE_GNT is set to 1b, the EE_CS output signal is mapped to the chip select of the EEPROM device.
EE_DI	2	0b	Data input to the EEPROM When EE_GNT is set to 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	3	X	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 read-only from a software perspective; writes to this bit has no effect.



FWE	5:4	01b	Flash Write Enable Control These two bits control whether or not writes to the Flash are allowed. 00b = Flash erase (along with bit 31 in the FLA register). 01b = Flash writes disabled. 10b = Flash writes enabled. 11b = Not allowed.
EE_REQ	6	0b	Request EEPROM Access Software must write a 1b to this bit to get direct EEPROM access. It has access when EE_GNT is set to 1b. When software completes the access, it must then write a 0b.
EE_GNT	7	0b	Grant EEPROM Access When this bit is set to 1b, software can access the EEPROM using the EE_SK, EE_CS, EE_DI, and EE_DO bits. This field is read-only.
EE_PRES	8	(See Description)	EEPROM Present Setting this bit to 1b indicates that an EEPROM is present and has the correct signature field. This field is read-only.
Auto_RD	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 field is not valid. This field is read-only.
EE_ADDR_SIZE	10	0b	EEPROM Address Size This field defines the address size of the EEPROM: 0b = 8- or 9-bit addresses. 1b = 16-bit address. This field is read-only.
EE_Size	14:11	0010b[1]	EEPROM Size This field defines the size of the EEPROM (see Table 4-4). This field is read-only.
Field	**Bit(s)**	**Initial Value**	**Description**
PCI _ANA_done	15	0b	PCIe Analog Done When set to 1b, indicates that the PCIe analog section read from EEPROM is done. This bit is cleared when auto-read starts. This bit is also set when the EEPROM is not present or when its signature field is not valid.
PCI _Core_ done	16	0b	PCIe Core Done When set to 1b, indicates that the core analog section read from EEPROM is done. This bit is cleared when auto-read starts. This bit is also set when the EEPROM is not present or when its signature field is not valid. This field is read-only. **Note:** This bit returns the relevant done indication for the function that reads the register.
PCI _ genarl _done	17	0b	PCIe General Done When set to 1b, indicates that the PCIe general section read from the EEPROM is done. This bit is cleared when auto-read starts. This bit is also set when the EEPROM is not present or when its signature field is not valid. This field is read-only.

PCI_ FUNC_ DONE	18	0b	PCIe Function Done When set to 1b, indicates that the PCIe function section read from EEPROM is done. This bit is cleared when auto-read starts. This bit is also set when the EEPROM is not present or when its signature field is not valid. This field is read-only. **Note**: This bit returns the relevant done indication for the function that reads the register.
CORE_ DONE	19	0b	Core Done When set to 1b, indicates that the core section read from the EEPROM is done. This bit is cleared when auto-read starts. This bit is also set when the EEPROM is not present or when its signature field is not valid. This field is read-only. **Note:** This bit returns the relevant done indication for the function that reads the register.
CORE_ CSR_ DONE	20	0b	Core CSR Done When set to 1b, indicates that the core CSR section read from the EEPROM is done. This bit is cleared when auto-read starts. This bit is also set when the EEPROM is not present or when its signature field is not valid. This field is read-only. **Note**: This bit returns the relevant done indication for the function that reads the register.
MAC_ DONE	21	0b	MAC Done When set to 1b, indicates that the MAC section read from the EEPROM is done. This bit is cleared when auto-read starts. This bit is also set when the EEPROM is not present or when its signature field is not valid. This field is read-only. **Note:** This bit returns the relevant done indication for the function that reads the register.
Reserved	31:22	0x0	Reserved Reads as 0b.

1. These bits are read from the EEPROM.

Table 4-4. EEPROM Sizes (Bits 14:11)

Field Value	EEPROM Size	EEPROM Address Size
0000b	128 Bytes	1 Byte
0001b	256 Bytes	1 Byte
0010b	512 Bytes	1 Byte
0011b	1 kB	2 Bytes
0100b	2 kB	2 Bytes
0101b	4 kB	2 Bytes
0110b	8 kB	2 Bytes
0111b	16 kB	2 Bytes
1000b	32 kB	2 Bytes
1001b:1111b	Reserved	Reserved

This register provides software-direct access to the EEPROM. Software controls the EEPROM by successive writes to the register. Data and address information is clocked into the EEPROM by software toggling the EESK bit (bit 2) of this register with EE_CS set to 1b. Data output from the EEPROM is latched into bit 3 of this register via the internal 62.5 MHz clock and is accessed by software via reads of this register.

Writes to the Flash device when writes are disabled (FWE = 01b) should not be attempted. Behavior after such an operation is undefined.

4.4.3.2.2 EEPROM Read Register – EERD (0x10014; RW)

Field	Bit(s)	Initial Value	Description
START	0	0b	Start Read Writing a 1b to this bit causes the EEPROM to read a 16-bit word at the address stored in the EE_ADDR field and then stores the result in the EE_DATA field. This bit is self-clearing
DONE	1	0b	Read Done Set this bit to 1b when the EEPROM read completes. Set this bit to 0b when the EEPROM read is in progress. Note that writes by software are ignored.
ADDR	15:2	0x0	Read Address This field is written by software along with *Start Read* to indicate that the address of the word to read.
DATA	31:16	0x0	Read Data Data returned from the EEPROM read.

This register is used by software 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 82598 reads the word from 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 using the value in the Read Data field.

When this register is used to read a word from the EEPROM, that word is not written to any of the 82598's internal registers even if it is normally a hardware-accessed word.

4.4.3.2.3 Flash Access Register – FLA (0x1001C; RW)

Field	Bit(s)	Initial Value	Description
FL_SCK	0	0b	Flash Clock Input 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.
FL_CE	1	0b	Flash Chip Select When FL_GNT is set to 1b, the FL_CE output signal is mapped to the chip select of the Flash device. Software enables the Flash by writing a 0b to this bit.
FL_SI	2	0b	Flash Data Input 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.

FL_SO	3	X	Flash Data Output 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.
FL_REQ	4	0b	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.
FL_GNT	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.
Reserved	29:6	0b	Reserved Reads as 0b.
FL_BUSY	30	0b	Flash Busy This bit is set to 1b while a write or an erase to the Flash is in progress, While this bit is cleared (reads as 0b), software can access to write a new byte to the Flash. **Note:** This bit is read-only from a software perspective.
FL_ER	31	0b	Flash Erase Command This command is sent to the Flash only if bits 5:4 of register EEC are also set to 00b. This bit is auto-cleared and reads as 0b.

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 (0) of this register with FL_CE set to 1b. 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.

In the 82598, the FLA register is only reset at Internal Power On Reset or LAN_PWR_GOOD (as opposed to legacy devices at software reset).

4.4.3.2.4 Manageability EEPROM Control Register – EEMNGCTL (0x10110; RW)

This register can be read/written by manageability firmware and is read-only to host software.

Field	Bit(s)	Initial Value	Description
ADDR	14:0	0x0	Address This field is written by manageability along with *Start Read* or *Start Write* to indicate which EEPROM 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.
WRITE	16	0b	Write This bit signals the EEPROM if the current operation is read or write. 0b = Read. 1b = Write.
EEBUSY	17	0b	EPROM Busy This bit indicates that the EEPROM is busy processing an EEPROM transaction and should not be accessed.

CFG_DONE	18	0b	Manageability Configuration Cycle is Complete This bit indicates that the manageability configuration cycle (configuration of PCIe and core) is complete. This bit is set to 1b by manageability firmware to indicate configuration is complete and cleared by hardware on any of the reset sources that caused the firmware to initialize the PHY. Writing a 0b by firmware does not affect the state of this bit. **Note:** Software should not try to access the PHY for configuration before this bit is set.
Reserved	30:19	0x0	Reserved
DONE	31	1b	Transaction Done This bit is cleared after the *Start Write* or *Start Read* bit is set by manageability and is set back again when the EEPROM write or read transaction completes.

4.4.3.2.5 Manageability EEPROM Read/Write Data – EEMNGDATA (0x10114; RW)

This register can be read/written by manageability firmware and is read-only to host software.

Field	Bit(s)	Initial Value	Description
WRDATA	15:0	0x0	Write Data Data to be written to the EEPROM.
RDDATA	31:16	X	Read Data Data returned from the EEPROM read. **Note:** This field is read only.

4.4.3.2.6 Manageability Flash Control Register – FLMNGCTL (0x10118; RW)

This register can be read/written by manageability firmware and is read-only to host software.

Field	Bit(s)	Initial Value	Description
ADDR	23:0	0x0	Address This field is written by manageability along with *Start Read* or *Start Write* to indicate which Flash address to read or write.
CMD	24:25	00b	Command Indicates which command should be executed. Valid only when the *CMDV* bit is set. 00b = Read command. 01b = Write command. 10b = Sector erase. **Note:** Sector erase is applicable only for Atmel* Flashes. 11b = Erase.
CMDV	26	0b	Command Valid When set, indicates that the manageability firmware issues a new command and is cleared by hardware at the end of the command.
FLBUSY	27	0b	Flash Busy This bit indicates that the Flash is busy processing a Flash transaction and should not be accessed.

Reserved	29:28	00b	Reserved
DONE	30	1b	Read Done This bit clears after the *CMDV* bit is set by manageability and is set back again when a Flash single-read transaction completes. When reading a burst transaction, this bit is cleared each time manageability reads FLMNGRDDATA.
WRDONE	31	1b	Global Done This bit clears after the *CMDV* bit is set by manageability and is set back again when all Flash transactions complete. For example, the Flash device finished reading all the requested read or other accesses (write and erase).

4.4.3.2.7 Manageability Flash Read Data – FLMNGDATA (0x1011C; RW)

This register can be read/written by manageability firmware and is read-only to host software.

Field	Bit(s)	Initial Value	Description
DATA	31:0	0x0	Read/Write Data On a read transaction, this register contains the data returned from the Flash read. On write transactions, bits 7:0 are written to the Flash.

4.4.3.2.8 Manageability Flash Read Counter – FLMNGCNT (0x10120; RW)

This register can be read/written by manageability firmware and is read-only to host software.

Field	Bit(s)	Initial Value	Description
Abort	31	0b	Abort Writing a 1b to this bit aborts the current burst read operation. It is also self-cleared by the Flash interface block when the Abort command executed.
Reserved	30:25	0x0	Reserved
RDCNT	24:0	0x0	Read Counter This counter holds the size of the Flash burst read in Dwords.

4.4.3.2.9 Flash Opcode Register – FLOP (0x01013C; RW)

This register enables the host or firmware to define the op-code used in order to erase a sector of the Flash or erase the entire Flash. This register is reset only at power on or during Internal Power On Reset or LAN_PWR_GOOD.

Note: Default values are applicable to Atmel* Serial Flash Memory devices.

Field	Bit(s)	Initial Value	Description
SERASE	7:0	0x52	Flash Block Erase Instruction The op-code for the Flash block erase instruction and is relevant only to Flash access by manageability.
DERASE	15:8	0x62	Flash Device Erase Instruction The op-code for the Flash erase instruction.
Reserved	31:16	0x0	Reserved

4.4.3.2.10 General Receive Control – GRC (0x10200; RW)

Field	Bit(s)	Initial Value	Description
MNG_EN	0	1b[1]	Manageability Enable This read-only bit indicates whether or not manageability functionality is enabled.
APME	1	0b[1]	Advance Power Management Enable If set to 1b, manageability wakeup is enabled. The 82598 sets the *PME_Status* bit in the Power Management Control/Status Register (PMCSR), asserts GIO_WAKE_N when manageability wakeup is enabled, and when it receives a matching magic packet. It is a single read/write bit in a single register, but has two values depending on the function that accesses the register.
Reserved	31:2	0x0	Reserved

1. Loaded from the EEPROM.

4.4.3.3 Interrupt Registers

4.4.3.3.1 Extended Interrupt Cause Register EICR (0x00800, RC)

Field	Bit(s)	Initial Value	Description
RTxQ	15:0	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 actual queue the appropriate RTxQ bit is according to the IVAR registers.
Reserved	19:16	0x0	Reserved
LSC	20	0b	Link Status Change This bit is set each time the link status changes (either from up to down or from down to up).
Reserved	21	0b	Reserved

MNG	22	0b	Manageability Event Detected Indicates that a manageability event happened. When the 82598 is in power down mode, the BMC might generate a PME for the same events that would cause an interrupt when the 82598 is in the D0 state.
Reserved	23	0b	Reserved
GPI_SDP0	24	0b	General Purpose Interrupt on SDP0 If GPI interrupt detection is enabled on this pin (via GPIE), this interrupt cause is set when the SDP0 is sampled high.
GPI_SDP1	25	0b	General Purpose Interrupt on SDP1 If GPI interrupt detection is enabled on this pin (via GPIE), this interrupt cause is set when the SDP1 is sampled high.
GPI_SDP2	26	0b	General Purpose Interrupt on SDP2 If GPI interrupt detection is enabled on this pin (via GPIE), this interrupt cause is set when the SDP2 is sampled high.
GPI_SDP3	27	0b	General Purpose Interrupt on SDP3 If GPI interrupt detection is enabled on this pin (via GPIE), this interrupt cause is set when the SDP3 is sampled high.
PBUR	28	0b	RX/TX Packet Buffer Unrecoverable Error This bit is set when an unrecoverable error is detected in the packet buffer memory for Rx or Tx packet.
DHER	29	0b	RX/TX Descriptor Handler Error This bit is set when an unrecoverable error is detected in the descriptor handler memory for Rx or Tx descriptors.
TCP Timer	30	0b	TCP Timer Expired Activated when the TCP timer reaches its terminal count.
Reserved	31	0b	Reserved

This register contains frequent interrupt conditions applicable to the 82598. Each time an interrupt-causing event occurs, the corresponding interrupt bit is set. An interrupt is generated each time one of the bits in this register is set and the corresponding bit is enabled using the Extended Interrupt Mask Set/Read register. An interrupt can be delayed by selecting a bit in the Interrupt Throttling register.

Note: The software device driver cannot determine the interrupt cause by using the RxQ and TxQ bits:

- Receive descriptor write back, receive queue full, receive descriptor minimum threshold hit, dynamic interrupt moderation for Rx.
- Transmit descriptor write back.

Writing 1b to any bit in the register clears that bit. Writing a 0b to any bit has no effect on that bit.

All register bits are cleared on a register read if GPIE.OCD bit is cleared; if GPIE.OCD bit is set, then only bits 29:20 are cleared.

Auto-clear can be enabled for any or all of the bits in this register.

4.4.3.3.2 Extended Interrupt Cause Set Register EICS (0x00808, WO)

Field	Bit(s)	Initial Value	Description
RTxQ	15:0	0x0	Set corresponding EICR RTxQ interrupt condition.
Reserved	19:16	0x0	Reserved
LSC	20	0b	Set link status change interrupt.
Reserved	21	0b	Reserved
MNG	22	0b	Set manageability event interrupt.
Reserved	23	0b	Reserved
GPI_SDP0	24	0b	Set general purpose interrupt on SDP0.
GPI_SDP1	25	0b	Set general purpose interrupt on SDP1.
GPI_SDP2	26	0b	Set general purpose interrupt on SDP2.
GPI_SDP3	27	0b	Set general purpose interrupt on SDP3.
PBUR	28	0b	Set RX/TX packet buffer unrecoverable error interrupt.
DHER	29	0b	Set RX/TX descriptor handler error interrupt.
TCP Timer	30	0b	Set corresponding EICR TCP timer interrupt condition.
Reserved	31	0b	Reserved

Software uses this register to set an interrupt condition. Any bit written with a 1b sets the corresponding bit in the Extended Interrupt Cause register (see Section 4.4.3.3.1) and clears the relevant EITR register if GPIE.EIMEN is set. An immediate interrupt is then generated if a bit in this register is set and the corresponding interrupt is enabled using the Extended Interrupt Mask Set/Read register.

If GPIE.EIMEN is not set, then an interrupt generated by setting a bit in this register waits for EITR expiration.

Note: Bits written with 0b are unchanged.

4.4.3.3.3 Extended Interrupt Mask Set/Read Register EIMS (0x00880, RWS)

Field	Bit(s)	Initial Value	Description
RTxQ	15:0	0x0	Mask bit for corresponding EICR RTxQ interrupt condition.
Reserved	19:16	0x0	Reserved
LSC	20	0b	Mask link status change interrupt.
Reserved	21	0b	Reserved
MNG	22	0b	Mask manageability event interrupt.

Reserved	23	0b	Reserved
GPI_SDP0	24	0b	Mask general purpose interrupt on SDP0.
GPI_SDP1	25	0b	Mask general purpose interrupt on SDP1.
GPI_SDP2	26	0b	Mask general purpose interrupt on SDP2.
GPI_SDP3	27	0b	Mask general purpose interrupt on SDP3.
PBUR	28	0b	Mask RX/TX packet buffer unrecoverable error interrupt.
DHER	29	0b	Mask RX/TX descriptor handler error interrupt.
TCP Timer	30	0b	Mask bit for corresponding EICR TCP timer interrupt condition.
Reserved	31	0b	Reserved

Reading this register reveals which bits have an interrupt mask set. An interrupt in EICR is enabled if its mask bit is set to 1b and disabled if its mask bit is set to 0b. A PCI interrupt is generated each a bit in this register is set and the corresponding interrupt occurs (subject to throttling). The occurrence of an interrupt condition is reflected by having a bit set in the Extended Interrupt Cause Read register (see Section 4.4.3.3.1).

An interrupt might be enabled by writing a 1b to the corresponding mask bit location (as defined in the EICR register) in this register. Bits written with a 0b are unchanged. Thus, if software needs to disable a particular interrupt condition (previously enabled), it must write to the Extended Interrupt Mask Clear Register, rather than writing a 0b to a bit in this register.

4.4.3.3.4 Extended Interrupt Mask Clear Register EIMC (0x00888, WO)

Field	Bit(s)	Initial Value	Description
RTxQ	15:0	0x0	Mask bit for corresponding EICR RTxQ interrupt condition.
Reserved	19:16	0x0	Reserved
LSC	20	0b	Mask link status change interrupt.
Reserved	21	0b	Reserved
MNG	22	0b	Mask manageability event interrupt.
Reserved	23	0b	Reserved
GPI_SDP0	24	0b	Mask general purpose interrupt on SDP0.
GPI_SDP1	25	0b	Mask general purpose interrupt on SDP1.
GPI_SDP2	26	0b	Mask general purpose interrupt on SDP2.
GPI_SDP3	27	0b	Mask general purpose interrupt on SDP3.
PBUR	28	0b	Mask RX/TX packet buffer unrecoverable error interrupt.
DHER	29	0b	Mask RX/TX descriptor handler error interrupt.

TCP Timer	30	0b	Mask bit for corresponding EICR TCP timer interrupt condition.
Reserved	31	0b	Reserved

Software uses this register to disable an interrupt. Interrupts are presented to the bus interface only when the mask bit is 1b and the cause bit is 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 (see Section 4.4.3.3.1).

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). Bits written with 0b are unchanged.

This register provides software with a way to disable interrupts. Software disables a given interrupt by writing a 1b to the corresponding bit.

4.4.3.3.5 Extended Interrupt Auto Clear Register EIAC (0x00810, RW)

Field	Bit(s)	Initial Value	Description
RTxQ	15:0	0x0	Auto-clear bit for corresponding EICR RTxQ interrupt condition.
Reserved	19:16	0x0	Reserved
LSC	20	0b	Auto-clear link status change interrupt.
Reserved	21	0b	Reserved
MNG	22	0b	Auto-clear manageability event interrupt
Reserved	23	0b	Reserved
GPI_SDP0	24	0b	Auto-clear general purpose interrupt on SDP0.
GPI_SDP1	25	0b	Auto-clear general purpose interrupt on SDP1.
GPI_SDP2	26	0b	Auto-clear general purpose interrupt on SDP2.
GPI_SDP3	27	0b	Auto-clear general purpose interrupt on SDP3.
PBUR	28	0b	Auto-clear RX/TX packet buffer unrecoverable error interrupt.
DHER	29	0b	Auto-clear RX/TX descriptor handler error interrupt.
TCP Timer	30	0b	Auto-clear bit for corresponding EICR TCP timer interrupt condition.
Reserved	31	0b	Reserved

This register is mapped like previous interrupt registers; each bit is mapped to a corresponding bit in the EICR. EICR bits that have auto-clear set are cleared when the MSI-X message that they trigger is sent on the PCIe bus. Note that an MSI-X message might be delayed by ITR moderation (from the time the EICR bit is activated). Bits without auto-clear set need to be cleared using a write-to-clear.

Read-to-clear is not compatible with auto-clear; if any bits are set to auto-clear, read-to-clear should be disabled (use the configuration register bit).

Bits 29:20 should never be set to auto clear since they share the same MSI-X vector.

4.4.3.3.6 Extended Interrupt Auto Mask Enable Register – EIAM (0x00890, RW)

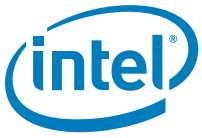
Each bit in this register enables the setting of the corresponding bit in the EIMC register following a write-to-clear to the EICR register or the setting of the corresponding bit in the EIMS register following a write-to-set to the EICS register.

Field	Bit(s)	Initial Value	Description
RTxQ	15:0	0x0	Auto-mask bit for corresponding EICR RTxQ interrupt condition.
Reserved	19:16	0x0	Reserved
LSC	20	0b	Auto-mask link status change interrupt.
Reserved	21	0b	Reserved
MNG	22	0b	Auto-mask manageability event interrupt.
Reserved	23	0b	Reserved
GPI_SDP0	24	0b	Auto-mask general purpose interrupt on SDP0.
GPI_SDP1	25	0b	Auto-mask general purpose interrupt on SDP1.
GPI_SDP2	26	0b	Auto-mask general purpose interrupt on SDP2.
GPI_SDP3	27	0b	Auto-mask general purpose interrupt on SDP3.
PBUR	28	0b	Auto-mask RX/TX packet buffer unrecoverable error interrupt.
DHER	29	0b	Auto-mask RX/TX descriptor handler error interrupt.
TCP Timer	30	0b	Auto-mask bit for corresponding EICR TCP timer interrupt condition.
Reserved	31	0b	Reserved

4.4.3.3.7 Extended Interrupt Throttle Registers – EITR (0x00820 – 0x0086C, RW)

Each ITR is responsible for an interrupt cause. The allocation of ITR to interrupt cause is through MSI-X allocation registers.

Field	Bit(s)	Initial Value	Description
Interval	15:0	0x0	Minimum Inter-interrupt Interval The interval is specified in 256 ns increments. Zero disables interrupt throttling logic.
Counter	31:16	Start	Down 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. This counter can be directly written by software at any time to alter the throttles performance.



Software uses this register to even out the delivery of interrupts to the host CPU. The register provides a guaranteed inter-interrupt delay between interrupts, regardless of network traffic conditions. To independently validate configuration settings, software should use the following algorithm to convert the inter-interrupt interval value to the common interrupts/seconds performance metric:

$$\text{interrupts/sec} = (256\ 10^{-9}\text{sec} \times \text{interval})^{-1}$$

For example, if the interval is programmed to 500d, the 82598 guarantees the CPU is not interrupted by it for 128 µs from the last interrupt. The maximum observed interrupt rate from the 82598 should never exceed 7813 interrupts/seconds.

Inversely, the inter-interrupt interval value can be calculated as:

$$\text{inter-interrupt interval} = (256\ 10^{-9}\text{sec} \times \text{interrupts/sec})^{-1}$$

The optimal performance setting for this register is system and configuration specific.

4.4.3.3.8 Interrupt Vector Allocation Registers IVAR (0x00900 + 4*n [n=0…24], RW)

These registers have two modes of operation:

1. In MSI-X mode, these registers define allocation of different interrupt causes to MSI-X vectors. Each INT_Alloc[i] (i=0…97) 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/Tx queue interrupt causes to one of the RTxQ bits in the EICR. Each INT_Alloc[i] (i=0…97) field is a byte indexing the appropriate RTxQ bit.

31 ….24	23 16	15 8	7 0
INT_Alloc[3]	INT_Alloc[2]	INT_Alloc[1]	INT_Alloc[0]
…	…	…	…

…	…	…	…
Reserved	Reserved	INT_Alloc[97]	INT_Alloc[96]

Field	Bit(s)	Initial Value	Description
INT_Alloc[0]	4:0	0x0	Defines the MSI-X vector assigned to the interrupt cause associated with this entry.
Reserved	6:5	0x0	Reserved
INT_Alloc _val[0]	7	0b	Valid bit for INT_Alloc[0]
INT_Alloc[1]	12:8	0x0	Defines the MSI-X vector assigned to the interrupt cause associated with this entry, as defined in.
Reserved	14:13	0x0	Reserved
INT_Alloc _val[1]	15	0b	Valid bit for INT_Alloc[1]
INT_Alloc[2]	20:16	0x0	Defines the MSI-X vector assigned to the interrupt cause associated with this entry, as defined in.

Reserved	22:21	0x0	Reserved
INT_Alloc _val[2]	23	0b	Valid bit for INT_Alloc[2]
INT_Alloc[3]	28:24	0x0	Defines the MSI-X vector assigned to the interrupt cause associated with this entry, as defined in.
Reserved	30:29	0x0	Reserved
INT_Alloc _val[3]	31	0b	Valid bit for INT_Alloc[3]

4.4.3.3.9 MSI-X Table - MSIXT (BAR3: 0x00000 – 0x0013C, RW)

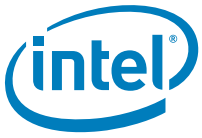
DWORD3	DWORD2	DWORD1	DWORD0	Entry	Address
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 0	0x00000
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 1	0x00010
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 2	0x00020
…	…	…	…	…	
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry (19)	0x00130

4.4.3.3.10 MSI-X Pending Bit Array – MSIXPBA (BAR3: 0x02000, RO)

Field	Bit(s)	Initial Value	Description
PENBIT	19:0	0x0	MSI-X Pending Bits Each bit is set to 1b when the appropriate interrupt request is set and cleared to 0b when the appropriate interrupt request is cleared.
Reserved	31:20	0x0	Reserved

4.4.3.3.11 MSI-X Pending Bit Array Clear – PBACL (0x11068, RW)

Field	Bit(s)	Initial Value	Description
PENBITCLR	19:0	0x0	MSI-X Pending Bits Clear Writing 1b to any bit clears it's content; writing 0b has no effect. Reading this register returns the MSIPBA.PENBIT value.
Reserved	31:20	0x0	Reserved



4.4.3.3.12 General Purpose Interrupt Enable – GPIE (0x00898, RW)

Field	Bit(s)	Initial Value	Description
SDP0_GPIEN	0	0b	General Purpose Interrupt Detection Enable for SDP0 If software-controllable IO pin SDP0 is configured as an input, this bit (when 1b) enables use for GPI interrupt detection.
SDP1_GPIEN	1	0b	General Purpose Interrupt Detection Enable for SDP1 If software-controllable IO pin SDP1 is configured as an input, this bit (when 1b) enables use for GPI interrupt detection.
SDP2_GPIEN	2	0b	General Purpose Interrupt Detection Enable for SDP2 If software-controllable IO pin SDP2 is configured as an input, this bit (when 1b) enables use for GPI interrupt detection.
SDP3_GPIEN	3	0b	General Purpose Interrupt Detection Enable for SDP3 If software-controllable IO pin SDP3 is configured as an input, this bit (when 1b) enables use for GPI interrupt detection.
MSIX_MODE	4	0b	MSIX Mode 0b = non-MSIX, IVAR map Rx/Tx causes to 16 EICR bits, but MSIX[0] is asserted for all. 1b = MSIX mode, IVAR maps Rx/Tx causes to 16 EICR bits.
OCD	5	0b	Other Clear Disable When set indicates that only bits 20-29 of the EICR are cleared on read.
EIMEN	6	0b	EICS Immediate Interrupt Enable When set, setting bit in the EICS causes an immediate interrupt. If not set, the EICS interrupt waits for EITR expiration
Reserved	29:5	0x0	Reserved
EIAME	30	0b	Extended Interrupt Auto Mask Enable When set (usually in MSI-X mode); upon initializing an MSI-X message, bits set in EIAM associated with this message is cleared. Otherwise, EIAM is used only after a read or write of the EICR/EICS registers.
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 82598 behaves in a way supporting legacy INT-x interrupts. **Note:** Should be cleared when working in INT-x or MSI mode and set in MSI-X mode.

The 82598 allows for up to four externally controlled interrupts. The lower four software-definable pins, SDP[3:0], can be mapped for use as GPI interrupt bits. The mappings are enabled by the SDPx_GPIEN bits only when these signals are also configured as inputs using 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 the following table for clarity.

Table 4-5. GPI to SDP Bit Mappings

SDP pin to be used as GPI	ESDP Field Settings		Resulting EICR bit (GPI)
	Directionality	Enable as GPI interrupt	
3	SDP3_IODIR	SDP3_GPIEN	27
2	SDP2_IODIR	SDP2_GPIEN	26
1	SDP1_IODIR	SDP1_GPIEN	25
0	SDP0_IODIR	SDP0_GPIEN	24

4.4.3.4 Flow Control Registers Description

4.4.3.4.1 Priority Flow Control Type Opcode – PFCTOP (0x03008; RW)

Field	Bit(s)	Initial Value	Description
FCT	15:0	0x8808	Class-Based Flow Control Type
FCOP	31:16	0x0101	Class-Based Flow Control Opcode

This register contains the *Type* field hardware that is matched against a recognized class-based flow control packet.

4.4.3.4.2 Flow Control Transmit Timer Value n – FCTTVn (0x03200 + 4*n[n=0..3]; RW)

Where each 32-bit register (n=0… 3) refers to two timer values (register 0 refers to timer 0 and 1, register 1 refers to timer 2 and 3, etc.).

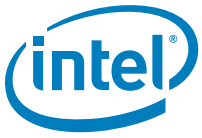
Field	Bit(s)	Initial Value	Description
TTV(2n)	15:0	0x0	Transmit Timer Value 2n Timer value included in XOFF frames as Timer (2n). For legacy 802.3X flow control packets, TTV0 is the only timer that is used.
TTV(2n+1)	31:16	0x0	Transmit Timer Value 2n+1 Timer value included in XOFF frames as Timer 2n+1.

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 (usually 64 bytes).

Note: The 82598 uses a fixed slot time value of 64 byte times.

4.4.3.4.3 Flow Control Receive Threshold Low – FCRTL (0x03220 + 8*n[n=0..7]; RW)

Where each 32-bit register (n=0… 7) refers to a different receive packet buffer.



Field	Bit(s)	Initial Value	Description
Reserved	3:0	0x0	Reserved The underlying bits might not be implemented in all versions of the 82598. Must be written with 0x0.
RTL[n]	18:4	0x0	Receive Threshold Low n Receive packet buffer n FIFO low water mark for flow control transmission (256 bytes granularity).
Reserved	30:19	0x0	Reserved Reads as 0x0. Should be written to 0x0 for future compatibility.
XONE[n]	31	0b	XON Enable n Per the receive packet buffer XON enable. 0b = Disabled 1b = Enabled.

This register contains the receive threshold used to determine when to send an XON packet and counts 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 incoming packets cross the receive high threshold (become more full), and then crosses the receive low threshold, with XONE enabled (1b), hardware transmits an XON frame.

Flow control reception/transmission is negotiated through by the auto negotiation process. When the 82598 is manually configured, flow control operation is determined by the RFCE and RPFCE bits.

4.4.3.4.4 Flow Control Receive Threshold High – FCRTH (0x03260 + 8*n[n=0..7]; RW)

Where each 32-bit register (n=0… 7) refers to a different receive packet buffer.

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0x0	Reserved The underlying bits might not be implemented in all versions of the 82598. Must be written with 0x0.
RTH[n]	18:4	0x0	Receive Threshold High n Receive packet buffer n FIFO high water mark for flow control transmission (16 bytes granularity).
Reserved	30:19	0x0	Reserved Reads as 0x0 Should be written to 0x0 for future compatibility.
FCEN[n]	31	0b	Flow control enable for receive packet buffer n.

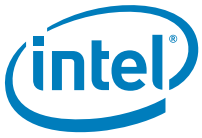
This register contains the receive threshold used to determine when to send an XOFF packet and counts in units of bytes. The value must be at least eight bytes less than the maximum number of bytes allocated to the receive packet buffer and the lower four bits must be programmed to 0x0 (16-byte granularity). 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.

4.4.3.4.5 Flow Control Refresh Threshold Value – FCRTV (0x032A0; RW)

Field	Bit(s)	Initial Value	Description
FC_refresh_th	15:0	0x0	Flow Control Refresh Threshold This value indicates the threshold value of the flow control shadow counter. When the counter reaches this value, and the conditions for a pause state are still valid (buffer fullness above low threshold value), a pause (XOFF) frame is sent to link partner.
Reserved	31:16	0x0	Reserved

4.4.3.4.6 Transmit Flow Control Status – TFCS (0x0CE00; RO)

Field	Bit(s)	Initial Value	Description
TXOFF	0	0b	Transmission Paused Pause state indication of the transmit function when symmetrical flow control is enabled.
Reserved	7:1	0x0	Reserved
TXOFF0	8	0b	Packet Buffer 0 Transmission Paused Pause state indication of the PB0 when class-based flow control is enabled.
TXOFF1	9	0b	Packet Buffer 1 Transmission Paused Pause state indication of the PB1 when class-based flow control is enabled.
TXOFF2	10	0b	Packet Buffer 2 Transmission Paused Pause state indication of the PB2 when class-based flow control is enabled.
TXOFF3	11	0b	Packet Buffer 3 Transmission Paused Pause state indication of the PB3 when class-based flow control is enabled.
TXOFF4	12	0b	Packet Buffer 4 Transmission Paused Pause state indication of the PB4 when class-based flow control is enabled.
TXOFF5	13	0b	Packet Buffer 5 Transmission Paused Pause state indication of the PB5 when class-based flow control is enabled.
TXOFF6	14	0b	Packet Buffer 6 Transmission Paused Pause state indication of the PB6 when class-based flow control is enabled.
TXOFF7	15	0b	Packet Buffer 7 Transmission Paused Pause state indication of the PB7 when class-based flow control is enabled.
Reserved	31:16	0x0	Reserved



4.4.3.5 Receive DMA Registers

4.4.3.5.1 Receive Descriptor Base Address Low – RDBAL (0x01000 + 0x40*n[n=0..63]; RW)

Field	Bit(s)	Initial Value	Description
0	6:0	0x0	Ignored on writes. Returns 0x0 on reads.
RDBAL	31:7	X	Receive Descriptor Base Address Low

This register contains the lower bits of the 64-bit descriptor base address. The lower seven bits are ignored. The receive descriptor base address must point to a 16-byte aligned block of data.

4.4.3.5.2 Receive Descriptor Base Address High – RDBAH (0x01004 + 0x40*n[n=0..63]; RW)

Field	Bit(s)	Initial Value	Description
RDBAH	31:0	X	Receive Descriptor Base Address [63:32]

This register contains the upper 32 bits of the 64-bit descriptor base address.

4.4.3.5.3 Receive Descriptor Length – RDLEN (0x01008 + 0x40*n[n=0..63]; RW)

Field	Bit(s)	Initial Value	Description
0	6:0	0x0	Ignore on write. Reads back as 0x0.
LEN	19:7	0x0	Descriptor Length.
Reserved	31:20	0x0	Reads as 0x0. Should be written to 0 for future compatibility.

This register sets the number of bytes allocated for descriptors in the circular descriptor buffer. It must be 128-byte aligned.

4.4.3.5.4 Receive Descriptor Head – RDH (0x01010 + 0x40*n[n=0..63]; RO)

Field	Bit(s)	Initial Value	Description
RDH	15:0	0x0	Receive Descriptor Head
Reserved	31:16	0x0	Reserved. Should be written with 0x0.

This register contains the head pointer for the receive descriptor buffer. The register points to a 16-byte datum. Hardware controls the pointer. The only time that software should write to this register is after a reset (hardware reset or CTRL.RST) and before enabling the receive function (RXCTRL.RXEN).

4.4.3.5.5 Receive Descriptor Tail – RDT (0x01018 + 0x40*n[n=0..63]; RW)

Field	Bit(s)	Initial Value	Description
RDT	15:0	0x0	Receive Descriptor Tail
Reserved	31:16	0x0	Reads as 0x0. Should be written to 0x0 for future compatibility.

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: If the 82598 uses the packet-split feature, software should write an even number to the tail register. The tail pointer should be set to point one descriptor beyond the last empty descriptor in host descriptor ring.

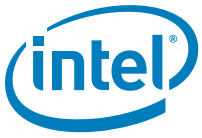
4.4.3.5.6 Receive Descriptor Control – RXDCTL (0x01028 + 0x40*n[n=0..63]; RW)

Field	Bit(s)	Initial Value	Description
PTHRESH	6:0	0x00	Pre-Fetch Threshold
Reserved	7	0x00	Reserved
HTHRESH	14:8	0x00	Host Threshold
Reserved	15	0x00	Reserved
WTHRESH	22:16	0x01	Write-Back Threshold
Reserved	24:23	0x00	Reserved
ENABLE	25	0b	Receive Queue Enable When set, the *Enable* bit enables the operation of the specific receive queue, upon read – get the actual status of the queue (internal indication that the queue is actually enabled/disabled).
Reserved	26	0b	Received
Reserved	31:27	0x00	Reserved

The register controls the fetching and write-back of receive descriptors. Three threshold values are used to determine when descriptors are read from and written to host memory.

PTHRESH is used to control when a pre-fetch of descriptors is considered. This threshold refers to the number of valid, unprocessed, receive descriptors in the on-chip descriptor buffer. If the number drops below PTHRESH, the algorithm considers pre-fetching descriptors from host memory. The host memory fetch does not happen, however, unless there are at least HTHRESH valid descriptors in host memory to fetch.

WTHRESH controls the write-back of processed receive descriptors. This threshold refers to the number of receive descriptors in the on-chip buffer which 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:

```
PTHRESH = 0..32
WTHRESH = 0..16
HTHRESH = 0, 4, 8
```

Note: For proper operation, the PTHRESH value should be larger than the number of buffers needed to accommodate a single packet/TSO.

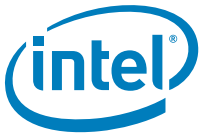
Note: Since the default value for write-back threshold is one, descriptors are normally written back as soon as one descriptor is available. WTHRESH must contain a non-zero value to take advantage of write-back bursting capabilities.

4.4.3.5.7 Split Receive Control Registers – SRRCTL (0x02100 – 0x0213C; RW)

Field	Bit(s)	Initial Value	Description
BSIZEPACKET	6:0	0x2	Receive Buffer Size for Packet Buffer The value is in 1 kB resolution. Value can be from 1 kB to 16 kB. Default buffer size is 2 kB. This field should not be set to 0x0. RXCTRL.DMBYPS should be set to 1b to bypass the descriptor monitor functionality.
Reserved	7	0b	Reserved. Should be written with 0b to ensure future compatibility.
BSIZEHEADER1	13:8	0x4	Receive Buffer Size for Header Buffer The value is in 64 bytes resolution. Value can be from 64 bytes to 1024 bytes. Default buffer size is 256 bytes. This field must be greater than zero if the value of DESCTYPE is greater or equal to two. Values above 1024 bytes are reserved for internal use only. NOTE: BSIZEHEADER must be bigger than zero if DESCTYPE is equal to 010b, 011b 100b or 101b.
Reserved	24:14	0x0	Reserved
DESCTYPE	27:25	000b	Define the descriptor type in RX. 000b = legacy 001b = Advanced descriptor one buffer 010b = Advanced descriptor header splitting 011b = Reserved 100b = Reserved 101b = Advanced descriptor header splitting always use header buffer. 110b – 111b = Reserved.
Reserved	31:28	0x0	Reserved Should be written with 0x0 to ensure future compatibility.

4.4.3.5.8 Rx DCA Control Register – DCA_RXCTRL (0x02200 – 0x0223C; RW)

Field	Bit(s)	Initial Value	Description
CPUID	4:0	0x0	Physical ID In Front Side Bus (FS)B platforms, the software device driver, after discovering the physical CPU ID and CPU Bus ID, programs it into these bits for hardware to associate physical CPU and bus ID with the adequate RSS queue. Bits 2:1 are Target Agent ID, bit 3 is the Bus ID. Bits 2:0 are copied into bits 3:1 in the TAG field of the TLP headers of PCIe messages. In CSI platforms, the software device driver programs a value, based on the relevant APIC ID, corresponding to the adequate RSS queue. This value is copied in the 4:0 bits of the *DCA Preferences* field in TLP headers of PCIe messages.
RX Descriptor DCA EN	5	0b	Descriptor DCA EN When set, hardware enables DCA for all Rx descriptors written back into memory. When cleared, hardware does not enable DCA for descriptor write-backs.



RX Header DCA EN	6	0b	Rx Header DCA EN When set, hardware enables DCA for all received header buffers. When cleared, hardware does not enable DCA for Rx header.
Reserved	7	0b	Reserved
RXdescReadNSEn	8	0b	Rx Descriptor Read No-Snoop Enable This bit must be reset to 0b to ensure correct functionality (except if the software device driver can guarantee the data is present in the main memory before the DMA process occurs (the software device driver has written the data with a write-through instruction).
RXdescReadROEn	9	1b	Rx Descriptor Read Relax Order Enable
RXdescWBNSen	10	0b	Rx Descriptor Write Back No-Snoop Enable **Note:** This bit must be reset to 0b to ensure correct functionality of the descriptor write-back.
RXdescWBROen	11	0b (RO)	Rx Descriptor Write Back Relax Order Enable This bit must be 0b to allow correct functionality of the descriptors write-back.
RXdataWriteNSEn	12	1b	Rx Data Write No Snoop Enable When 0b, the last bit of the *Packet Buffer Address* field in advanced receive descriptor is used as least significant bit of the packet buffer address (A0), thus enabling 8-bit alignment of the buffer. When 1b, the last bit of the *Packet Buffer Address* field in advanced receive descriptor is used as *No-Snoop Enabling* (NSE) bit. In this case, the buffer is 16-bit aligned. In this case, (bit set to 1b), the NSE bit determines whether the data buffer is snooped or not.
RXdataWriteROEn	13	1b	Rx Data Write Relax Order Enable
RxRepHeaderNSEn	14	0b	Rx Split Header No-Snoop Enable This bit must be reset to 0b to enable correct functionality of a header write to host memory.
RxRepHeaderROEn	15	1b	Rx Split Header Relax Order Enable
Reserved	31:16	0x0	Reserved

The Rx data write no-snoop is activated when the *NSE* bit is set in the receive descriptor.

4.4.3.5.9 Receive DMA Control Register – RDRXCTL (0x02F00; RW)

Field	Bit(s)	Initial Value	Description
RDMTS	1:0	00b	Receive Descriptor Minimum Threshold Size The corresponding interrupt is set each time the fractional number of free descriptors becomes equal to RDMTS. 00b = 1/2. 01b = 1/4. 10b = 1/8. 11b = Reserved.
Reserved	2	0b	Reserved
DMAIDONE	3	0b	DMA Init Done When read as 1b, indicates that the DMA init cycle is done (RO).

Reserved	4	0b	Reserved
MVMEN	5	0b	DMA Configuration for MAC/VLAN (VMDq) Mode Registers Mapping This mode is enabled when set to 1b.
MCEN	6	0b	DMA Configuration for Multiple Cores (RSS) Registers Mapping This mode is enabled when set to 1b.
Reserved	7	00x	Reserved
RxDPipeSize	12:8	0x0	Receive Data Pipe Size Limits the amount of pending bytes in the DMA Rx queue (resolution in 1 kB).
Reserved	31:13	0x0	Reserved

4.4.3.5.10 Receive Packet Buffer Size – RXPBSIZE (0x03C00 – 0x03C1C; RW)

Field	Bit(s)	Initial Value	Description
Reserved	9:0	0x0	Reserved
SIZE	19:10	0x200/0	Receive Packet buffer size Default values: 0x200 (512 kB) for RXPBSIZE0. 0x0 (0 kB) for RXPBSIZE1-7. Other than the default configuration of one packet buffer, the 82598 supports two more configurations: Partitioned receive equal: 0x40 (64 kB) for RXPBSIZE0-7. Partitioned receive not equal: 0x50 (80 kB) for RXPBSIZE0-3. 0x30 (48 kB) for RXPBSIZE4-7.
Reserved	31:20	0x0	Reserved

4.4.3.5.11 Receive Control Register – RXCTRL (0x03000; RW)

Field	Bit(s)	Initial Value	Description
RXEN	0	0b	Receive Enable When set to 0b, filter inputs to the packet buffer are ignored.
DMBYPS	1	1b	Descriptor Monitor Bypass When set to 1b, the descriptor monitor (checking if there are enough descriptors in the target queue) is disabled.
Reserved	31:2	0x0	Reserved

4.4.3.5.12 Drop Enable Control – DROPEN (0x03D04 – 0x03D08; RW)

Field	Bit(s)	Initial Value	Description
Drop_En	31:0	0x0	Drop Enabled If set to 1b, packets received to the queue when no descriptors are available to store them are dropped. If set to 0b, packets received to the queue when no descriptors are available to store are held in an internal buffer until descriptors are available again. Each bit represents the appropriate queue (bit 0 in DROPEN0 represents queue0; bit 31 in DROPEN1 represents queue 63). When RXCTRL.DMBYPS is set to 1b, only packets received to a disabled queue are dropped.”

4.4.3.6 Receive Registers

4.4.3.6.1 Receive Checksum Control – RXCSUM (0x05000; RW)

Field	Bit(s)	Initial Value	Description
Reserved	11:0	0x0	Reserved
IPPCSE	12	0b	IP Payload Checksum Enable
PCSD	13	0b	RSS/Fragment Checksum Status Selection When set to 1b, the extended descriptor write-back has the RSS field. When set to 0b, it contains the fragment checksum.
Reserved	31:14	0x0	Reserved

The Receive Checksum Control register controls receive checksum offloading features. The 82598 supports offloading of three receive checksum calculations: the fragment checksum, the IP header checksum, and the TCP/UDP checksum.

PCSD

The *Fragment Checksum* and *IP Identification* fields are mutually exclusive with the RSS hash. Only one of the two options is reported in the Rx descriptor. The RXCSUM.PCSD affect is listed in the following table:

RXCSUM.PCSD	0 (Checksum Enable)	1 (Checksum Disable)
	Fragment checksum and IP identification are reported in the Rx descriptor	RSS hash value is reported in the Rx descriptor

IPPCSE

This is the IPPCSE control the fragment checksum calculation. As previously noted, the fragment checksum shares the same location as the RSS field. The fragment checksum is reported in the receive descriptor when the RXCSUM.PCSD bit is cleared.

If RXCSUM.IPPCSE cleared (the default value), the checksum calculation is not done and the value that is reported in the Rx fragment checksum field is 0b.

If the RXCSUM.IPPCSE is set, the fragment checksum is aimed to accelerate checksum calculation of fragmented UDP packets.

This register should only be initialized (written) when the receiver is not enabled (only write this register when RXCTRL.RXEN = 0b).

4.4.3.6.2 Receive Filter Control Register – RFCTL (0x05008; RW)

Field	Bit(s)	Initial Value	Description
Reserved	5:0	0x0	Reserved
Reserved	6	0b	Reserved. Must be set to 0b.
Reserved	7	0b	Reserved. Must be set to 0b.
NFS_VER	9:8	0x0	NFS Version 00b = NFS version 2. 01b = NFS version 3. 10b = NFS version 4. 11b = Reserved for future use.
Reserved	10	0b	Reserved. Must be set to 0b.
Reserved	11	0b	Reserved. Must be set to 0b.
Reserved	13:12	00b	Reserved
Reserved	14	0b	Reserved. Must be set to 0b.
Reserved	15	00b	Reserved
Reserved	16	0b	Reserved. Must be set to 0b.
Field	**Bit(s)**	**Initial Value**	**Description**
Reserved	17	0b	Reserved. Must be set to 0b.
Reserved	31:18	0x0	Reserved. Should be written with 0x0 to ensure future compatibility.

4.4.3.6.3 Multicast Table Array – MTA (0x05200-0x053FC; RW)

Field	Bit(s)	Initial Value	Description
Bit Vector	31:0	X	Word wide bit vector specifying 32 bits in the multicast address filter table.

The 82598 provides multicast filtering for 4096 multicast addresses by providing a single bit entry per multicast address. The 4096 address locations are organized in a multicast table array – 128 registers of 32 bits.

Only 12 bits out of the 48-bit destination address are considered as multicast addresses. The 12 bits can be selected by the *MO* field of the MCSTCTRL register.

Figure 4-1 shows the multicast lookup algorithm. The destination address shown represents the internally stored ordering of the received DA. Note that bit 0 is the first bit on the wire.

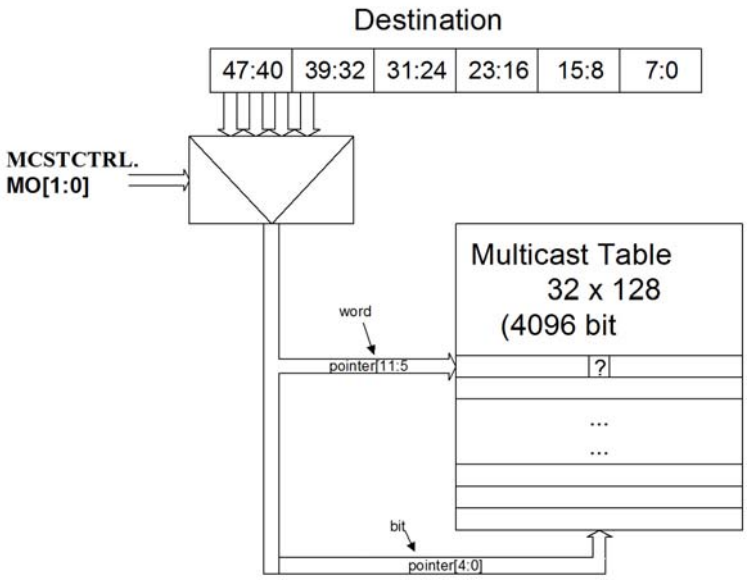


Figure 4-1. Multicast Lookup Algorithm

4.4.3.6.4 Receive Address Low – RAL (0x05400 + 8*n[n=0..15]; RW)

While "n" is the exact unicast/multicast address entry and it is equals to 0,1,…15.

Field	Bit(s)	Initial Value	Description
RAL	31:0	X	Receive Address Low The lower 32 bits of the 48-bit Ethernet address.

These registers contain the lower bits of the 48-bit Ethernet address. All 32 bits are valid. If the EEPROM is present, the first register (RAL0) is loaded from the EEPROM. The RAL value should be configured to the register in host order.

4.4.3.6.5 Receive Address High – RAH (0x05404 + 8*n[n=0..15]; RW)

While "n" is the exact unicast/multicast address entry and it is equals to 0,1,…15.

Field	Bit(s)	Initial Value	Description
RAH	15:0	X	Receive Address High The upper 16 bits of the 48-bit Ethernet address.
Reserved	17:16	00b	Reserved
VIND	21:18	0x0	VMDq output index Defines the VMDq output index associated with a received packet that matches this MAC address (RAH and RAL).
Reserved	30:22	0x0	Reserved. Reads as 0. Ignored on write.
AV	31	See Description	Address Valid Cleared after master reset. If the EEPROM is present, the *Address Valid* field of Receive Address register 0 is set to 1b after a software or PCI reset or EEPROM read. In entries 0-15 this bit is cleared by master reset.

The above registers contain the upper bits of a 48-bit Ethernet address. The complete address is (RAH, RAL; for all 16 register pairs). AV determines whether this address is compared against the incoming packet. AV is cleared by a master reset.

Note: The first Receive Address register (RAR0) is also used for exact match pause frame checking (DA matches the first register). RAR0 should always be used to store the Ethernet MAC address of the 82598.

After reset, if an EEPROM is present, the first register (Receive Address register 0) is loaded from the IA field in the EEPROM, its Address Select field is 00b, and its Address Valid field is 1b. If no EEPROM is present, the Address Valid field is 0b. The Address Valid field for all of the other registers are 0b.

The RAH value should be configured to the register in host order.

4.4.3.6.6 Packet Split Receive Type Register – PSRTYPE (0x05480 – 0x054BC, RW)

Field	Bit(s)	Initial Value	Description
PSR_type0	0	0b	Reserved
PSR_type1	1	1b	Header includes MAC, (VLAN/SNAP) 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, 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), 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.

PSR_type10	10	0b	Reserved
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.
PSR_type13	13	0b	Reserved
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.
PSR_type16	16	0b	Reserved
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	X	Reserved

This bit mask table enables or disables each type of header to be split.

4.4.3.6.7 VLAN Filter Table Array – VFTA (0x0A000-0x0A9FC; RW)

The VLAN Filter Table Array structure is shown in Figure 4-2. Each of five sections has 128 lines, each a Dword wide. The first section contains 128 lines of 32-bits that create a 4096-bit long VLAN filter. Each bit corresponds to one value of the 12-bit VLAN tag.

The next four sections contain the VMDq output index for VLAN tag values contained in the first section, (the first section contains VMDq outputs for each of the first bytes in the first section, the second section contains VMDq outputs for each of the second bytes in the first section and so forth). The first byte in the first section has its VMDq values in the second section first Dword.

For example, bit 0 in section 1 of line 0 corresponds to a VLAN tag of 0x000. Bits 3:0 in section 2 of line 0 contain the VMDq output index for VLAN tag of 0x000. Bit 1 in section 1 of line 0 corresponds to a VLAN tag of 0x001. Bits 7:4 in section 2 of line 0 contain the VMDq output index for VLAN tag of 0x001, etc.

Note: All accesses to this table must be 32-bit.

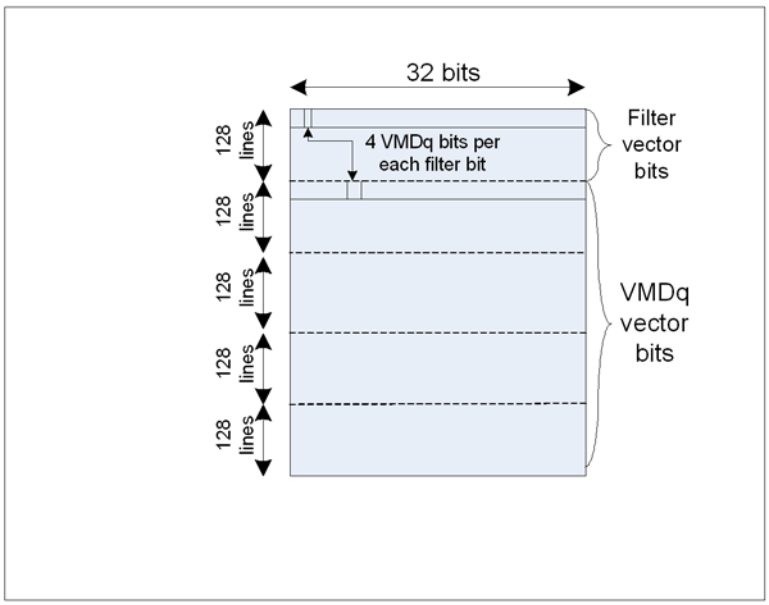


Figure 4-2. VLAN Filter Table Array – VFTA

The general structure of the VFTA memory is as follows.

Table 4-6. Structure of VFTA Memory

DW in line n	Bit(s)	Description
DW0	31:0	Filter value for VLAN tag value equal to [31:0] Each bit when set, enables packets with this VLAN tag value to pass. When cleared, blocks packets with this VLAN tag.
…		
DW127	31	Filter value for VLAN tag value equal to [4095:4064] Each bit when set, enables packets with this VLAN tag value to pass. When cleared, blocks packets with this VLAN tag.
DW128	3:0	VMDq output index for VLAN tag value 0x000.
DW128	7:4	VMDq output index for VLAN tag value 0x001.
…		
DW128	31:28	VMDq output index for VLAN tag value 0x007.
…		
DW in line n	**Bit(s)**	**Description**
DW255	31:28	VMDq output index for VLAN tag value 0xFE7.
DW256	3:0	VMDq output index for VLAN tag value 0x008.
DW256	7:4	VMDq output index for VLAN tag value 0x009.

Table 4-6. Structure of VFTA Memory

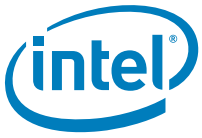
…		
DW256	31:28	VMDq output index for VLAN tag value 0x00F.
…		
DW383	31:28	VMDq output index for VLAN tag value 0xFEF.
DW384	3:0	VMDq output index for VLAN tag value 0x010.
DW384	7:4	VMDq output index for VLAN tag value 0x011.
…		
DW384	31:28	VMDq output index for VLAN tag value 0x017.
…		
DW511	31:28	VMDq output index for VLAN tag value 0xFF7.
DW512	3:0	VMDq output index for VLAN tag value 0x018.
DW512	7:4	VMDq output index for VLAN tag value 0x019.
…		
DW512	31:28	VMDq output index for VLAN tag value 0x01F.
…		
DW640	31:28	VMDq output index for VLAN tag value 0xFFF.

4.4.3.6.8 Filter Control Register – FCTRL (0x05080, RW)

Field	Bit(s)	Initial Value	Description
Reserved	31:16	0x0	Reserved
RFCE	15	0b	Receive Flow Control Enable Indicates that the 82598 responds to the reception of link flow control packets. If auto negotiation is enabled, this bit should be set by software to the negotiated flow control value. **Note:** When set, the 82598 does not count received flow control frames. **Note:** This bit should not be set if bit 14 is set.
RPFCE	14	0b	Receive Priority Flow Control Enable Indicates that the 82598 responds to the reception of priority flow control packets. If auto negotiation is enabled this bit should be set by software to the negotiated flow control value. **Note:** Receive priority flow control and receive link flow control are mutually exclusive and should not be configured at the same time. **Note:** This bit should not be set if bit 15 is set.
DPF	13	0b	Discard Pause Frame When set to 1b, unicast pause frames are sent to the host. Setting this bit to 1b causes unicast pause frames to be discarded only when RFCE or RPFCE are set to 1b. If both RFCE and RPFCE are set to 0b, this bit has no effect on incoming pause frames.

PMCF	12	0b	Pass MAC Control Frames Filter out unrecognized pause (flow control opcode does not match) and other control frames. 0b = Filter unrecognized pause frames. 1b = Pass/forward unrecognized pause frames.
Reserved	11	0b	Reserved
BAM	10	0b	Broadcast Accept Mode 0b – Ignore broadcast packets to host. 1b – accept broadcast packets to host.
UPE	9	0b	Unicast Promiscuous Enable 0b = Disabled. 1b = Enabled.
MPE	8	0b	Multicast Promiscuous Enable 0b = Disabled. 1b = Enabled.
Reserved	7:2	0x0	Reserved
Field	**Bit(s)**	**Initial Value**	**Description**
SBP	1	0b	Store Bad Packets 0b = Do not store. 1b = Store. Note that CRC errors before the SFD are ignored. Any packet must have a valid SFD (RX_DV with no RX_ER in the XGMII/GMII interface) in order to be recognized by the 82598 (even bad packets). **Note:** Packets with errors are not routed to manageability even if this bit is set. When this bit is set to 1b, it is not guaranteed that the status in the descriptor write-back is valid for packets shorter than 64 bytes. The queue assignment is not guaranteed. The relevant error bits are still valid. **Note:** Packets with a valid error (caused by a byte error or illegal error) might have data corruption in the last eight bytes when stored in host memory if the SBP bit is set.
Reserved	0	0b	Reserved

Note: Before receive filters are being updated/modified the RXCTRL.RXEN bit should be set to 0b. After the proper filters have been set the RXCTRL.RXEN bit can be set to 1b to re-enable the receiver.



4.4.3.6.9 VLAN Control Register – VLNCTRL (0x05088, RW)

Field	Bit(s)	Initial Value	Description
VME	31	0b	VLAN Mode Enable When set to 1b, on receive, VLAN information is stripped from 802.1q packets.
VFE	30	0b	VLAN Filter Enable 0b = Disabled (filter table does not decide packet acceptance). 1b = Enabled (filter table decides packet acceptance for 802.1q packets).
CFIEN	29	0b	Canonical Form Indicator Enable 0b = Disabled (CFI bit not compared to decide packet acceptance). 1b = Enabled (CFI from packet must match next CFI field to accept 802.1q packets).
CFI	28	0b	Canonical Form Indicator Bit Value If CFIEN is set to 1b, then 802.1q packets with CFI equal to this field are accepted; otherwise, the 802.1q packet is discarded.
Reserved	27:16	0x0	Reserved
VET	15:0	0x8100	VLAN Ether Type This register contains the type field that the hardware matches against to recognize an 802.1Q (VLAN) Ethernet packet. To be compliant with the 802.3ac standard, this register should be programmed with the value 0x8100. For VLAN transmission the upper byte is first on the wire (VLNCTRL.VET[15:8]).

4.4.3.6.10 Multicast Control Register – MCSTCTRL (0x05090, RW)

Field	Bit(s)	Initial Value	Description
Reserved	31:3	0x0	Reserved
MFE	2	0b	Multicast Filter Enable 0b = Disabled (filter is not applied – all multicast packets are not accepted). 1b = Enabled.
MO	1:0	00b	Multicast Offset This determines which bits of the incoming multicast address are used in looking up the bit vector. 00b = [47:36]. 01b = [46:35]. 10b = [45:34]. 11b = [43:32].

4.4.3.6.11 Multiple Receive Queues Command Register MRQC (0x05818; RW)

Field	Bit(s)	Initial Value	Description
RSS Enable	0	0b	RSS Enable When set, enables Receive Side Scaling (RSS) operation. 0b = RSS disabled. 1b = RSS enabled.

Reserved	15:1	0x0	Reserved
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] = Reserved. Bit[20] = Enable IPv6 hash function. Bit[21] = Enable TcpIPv6 hash function. Bit[22] = Enable UdpIPv4. Bit[23] = Enable UdpIPv6. Bit[24] = Enable UdpIPv6Ext. Bits[31:25] = Reserved (0x0).

Note: Disabling RSS on the fly is not allowed. Model usage is to reset the 82598 after disabling RSS.

4.4.3.6.12 VMDq Control Register – VMD_CTL (0x0581C; RW)

Field	Bit(s)	Initial Value	Description
VMDq Enable	0	0b	VMDq Enable When set, enables VMDq operation. 0b = VMDq disabled. 1b = VMDq enabled.
VMDq Filter	1	0b	VMDq Filter Determines the filtering mode used for VMDq filtering: 0b = MAC filtering. 1b = Reserved. This bit has no impact when VMDq is disabled.
Reserved	3:2	00b	Reserved
Default VMDq output index	7:4	0x0	Default VMDq output index Determines the VMDq output index for received packets that cannot be classified by the VMDq procedures (such as broadcast packets).
Reserved	31:8	0x0	Reserved

4.4.3.6.13 Immediate Interrupt Rx IMIR (0x05A80 + 4*n[n=0..7], RW)

This register defines the filtering that determines which packet triggers a dynamic interrupt moderation.

Field	Bit(s)	Initial Value	Description
PORT	15:0	0x0	Destination TCP Port This field is compared with the destination TCP port in incoming packets. The port value should be configured to the register in host order.

PORT_IM_EN	16	0b	Destination TCP Port Enable Allows issuing an immediate interrupt if all the following three conditions are met: • Packet TCP destination port is equal to *Port* field • Packet length of incoming packet is smaller than Size_Thresh in Im_IMIREXT register • At least one of the TCP control bits of incoming packets is set and the corresponding bit in the *CtrlBit* field in the IMIREXT register is set.
PORT_BP	17	0b	Port Bypass When 1b, the TCP port check is bypassed and only other conditions are checked. When 0b, the TCP port is checked to fit to *Port* field.
Reserved	31:18	0	Reserved

Another register includes a size threshold and a control bits bitmap to trigger an immediate interrupt.

4.4.3.6.14 Immediate Interrupt Rx Extended IMIREXT (0x05AA0 + 4*n[n=0..7], RW)

Field	Bit(s)	Initial Value	Description
Size_Thresh	11:0	0x0	Size Threshold These 12 bits define a size threshold; a packet with length below this threshold triggers an interrupt. Enabled by Size_Thresh_en.
Size_BP	12	0b	Size Bypass When 1b, the size check is bypassed. When 0b, the size check is performed.
CtrlBit	18:13	0x0	Control Bit When a bit in this field is equal to 1b, an interrupt is immediately issued after receiving a packet with corresponding TCP control bits turned on. Bit: 13: URG = Urgent pointer field significant. 14: ACK = Acknowledgment field. 15: PSH = Push function. 16: RST = Reset the connection. 17: SYN = Synchronize sequence numbers. 18: FIN = No more data from sender.
CtrlBit_BP	19	0b	Control Bits Bypass When 1b, the control bits check is bypassed. When 0b, the control bits check is performed.
Reserved	31:20	0x0	Reserved

4.4.3.6.15 Immediate Interrupt Rx VLAN Priority Register IMIRVP (0x05AC0, RW)

Field	Bit(s)	Initial Value	Description
Vlan_Pri	2:0	000b	VLAN Priority This field includes the VLAN priority threshold. When Vlan_pri_en is set to 1b, then an incoming packet with VLAN tag with a priority equal or higher to VlanPri triggers an immediate interrupt, regardless of the ITR moderation.

Vlan_pri_en	3	0b	VLAN Priority Enable When 1b, an incoming packet with VLAN tag with a priority equal or higher to Vlan_Pri triggers an immediate interrupt, regardless of the ITR moderation. When 0b, the interrupt is moderated by ITR.
Reserved	31:4	0x0	Reserved

4.4.3.6.16 Indirection Table – RETA (0x05C00-0x0057C; RW)

The indirection table is a 128-entry table, each entry is 8 bits wide. Each entry stores a 4-bit RSS output index or a pair of 4-bit indices. The table is configured through the following read/write registers.

31 24	23 16	15 8	7 0
Entry 3	Entry 2	Entry 1	Entry 0
		…	…

Entry 127	…	…	…

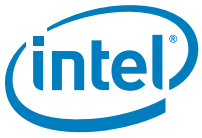
Field	Dword/ Bit(s)	Initial Value	Description
Entry0	7:0	0x0	Determines RSS output index or indices for hash value of 0x00.
Entry1	15:8	0x0	Determines RSS output index or indices for hash value of 0x01.
Entry2	23:16	0x0	Determines RSS output index or indices for hash value of 0x02.
Entry3	31:24	0x0	Determines RSS output index or indices for hash value of 0x03.

Each entry (byte) of the indirection table contains the following information:

- Bits [7:4] – RSS output index 1 (optional)
- Bits [3:0] – RSS output index 0

7:4	3:0
RSS index 1	RSS index 0

The contents of the indirection table is not defined following reset of the Memory Configuration registers. System software must initialize the table prior to enabling multiple receive queues. It might also 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.



If the operating system provides an indirection table whose size is smaller than 128 bytes, software should replicate the operating system-provided indirection table to span the entire 128 bytes of the hardware indirection table.

4.4.3.6.17 RSS Random Key Register – RSSRK (0x05C80-0x05CA4; RW)

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]

Field	Dword/ Bit(s)	Initial Value	Description
K0	7:0	0x0	Byte 0 of the RSS random key.
K1	15:8	0x0	Byte 1 of the RSS random key.
K2	23:16	0x0	Byte 2 of the RSS random key.
K3	31:24	0x0	Byte 3 of the RSS random key.

4.4.3.7 Transmit Register Descriptions

4.4.3.7.1 Transmit Descriptor Base Address Low – TDBAL (0x06000 + n*0x40[n=0..31]; RW)

Field	Bit(s)	Initial Value	Description
0	6:0	0x0	Ignored on writes. Returns 0x0 on reads.
TDBAL	31:7	X	Transmit Descriptor Base Address Low

This register contains the lower bits of the 64-bit descriptor base address. The lower seven bits are ignored. The transmit descriptor base address must point to a 16-byte aligned block of data.

4.4.3.7.2 Transmit Descriptor Base Address High – TDBAH (0x06004 + n*0x40[n=0..31]; RW)

Field	Bit(s)	Initial Value	Description
TDBAH	31:0	X	Transmit Descriptor Base Address [63:32]

This register contains the upper 32 bits of the 64-bit descriptor base address.

4.4.3.7.3 Transmit Descriptor Length – TDLEN (0x06008 + n*0x40[n=0..31]; RW)

Field	Bit(s)	Initial Value	Description
0	6:0	0x0	Ignore on write. Reads back as 0x0.
LEN	19:7	0x0	Descriptor Length
Reserved	31:20	0x0	Reads as 0x0. Should be written to 0x0.

This register contains the descriptor length and must be 128-byte aligned.

4.4.3.7.4 Transmit Descriptor Head – TDH (0x06010 + n*0x40[n=0..31]; RO)

Field	Bit(s)	Initial Value	Description
TDH	15:0	0x0	Transmit Descriptor Head
Reserved	31:16	0x0	Reserved. Should be written with 0x0.

This register contains the head pointer for the transmit descriptor ring. It points to a 16-byte datum. Hardware controls the pointer. The only time that software should write to this register is after a reset (hardware reset or CTRL.RST) and before enabling the transmit function (TXDCTL.ENABLE).

If software writes to this register while the transmit function is enabled, on-chip descriptor buffers might be invalidated and hardware behavior might be indeterminate.

4.4.3.7.5 Transmit Descriptor Tail – TDT (0x06018 + n*0x40[n=0..31]; RW)

Field	Bit(s)	Initial Value	Description
TDT	15:0	0x0	Transmit Descriptor Tail
Reserved	31:16	0x0	Reads as 0x0. Should be written to 0x0 for future compatibility.

This register contains the tail pointer for the transmit descriptor ring. It 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.

4.4.3.7.6 Transmit Descriptor Control – TXDCTL (0x06028 + n*0x40[n=0..31]; RW)

Field	Bit(s)	Initial Value	Description
PTHRESH	6:0	0x00	Pre-Fetch Threshold
Reserved	7	0x00	Reserved
HTHRESH	14:8	0x00	Host Threshold
Reserved	15	0x00	Reserved
WTHRESH	22:16	0x00	Write-Back Threshold
Reserved	24:23	0x00	Reserved
Enable	25	0b	Transmit Queue Enable When set, the *Enable* bit enables the operation of the specific transmit queue, upon read – get the actual status of the queue (internal indication that the queue is actually enabled/disable).
Reserved	31:26	0b	Reserved
Reserved	31:27	0x00	Reserved

This register controls the fetching and write-back of transmit descriptors. Three threshold values are used to determine when descriptors are read from and written to host memory.

PTHRESH is used to control when a pre-fetch of descriptors is considered. This threshold refers to the number of valid, unprocessed transmit descriptors the chip has in its on-chip buffer. If the number drops below PTHRESH, the algorithm considers pre-fetching descriptors from host memory. This fetch does not happen, however, unless there are at least HTHRESH valid descriptors in host memory to fetch.

WTHRESH 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.

Note: When WTHRESH = 0b, only descriptors with the *RS* bit set is written back.

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 the *RS* bit set. Furthermore, with transmit descriptor bursting enabled, some descriptors are written back that did not have the *RS* bit set in their respective descriptors.

For proper operation, the PTHRESH value should be larger than the number of buffers needed to accommodate a single packet/TSO.

Possible values:

- PTHRESH = 0..32
- WTHRESH = 0..16
- HTHRESH = 0, 4, 8

4.4.3.7.7 Tx Descriptor Completion Write Back Address Low – TDWBAL (0x06038 + n*0x40[n=0..31]; RW)

Field	Bit(s)	Initial Value	Description
Head_WB_En	0	0b	Head Write-Back Enable When 1b, head write-back is enabled. When 0b, head write-back is disabled.
Reserved	1	0b	Reserved
Reserved	3:2	00b	Reserved
HeadWB_Low	31:4	0x00	Lowest 32 bits of head write-back memory location (Dword-aligned). Last four bits are always 0000b.

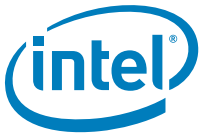
4.4.3.7.8 Tx Descriptor Completion Write Back Address High – TDWBAH (0x0603C + n*0x40[n=0..31]; RW)

Field	Bit(s)	Initial Value	Description
HeadWB_High	31:0	0x0000 0000	Highest 32 bits of head write-back memory location (for 64-bit addressing)

4.4.3.7.9 DMA TX Control – DTXCTL (0x07E00; RW)

This register controls whether or not the *IP Identification* field scrolls on 15-bit or 16- bit boundaries in TSO packets.

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved
Reserved	1	0b	Reserved
ENDBUBD	2	0b	Enable DBU buffer division, enable writing to DBU non-zero buffer.
Reserved	31:3	0x0	Reserved



4.4.3.7.10 Tx DCA Control Register – DCA_TXCTRL (0x07200 – 0x0723C; RW)

Field	Bit(s)	Initial Value	Description
CPUID	4:0	0x0	Physical ID In FSB platforms, the software device driver, upon discovery of the physical CPU ID and CPU Bus ID, programs it into these bits for hardware to associate Physical CPU and Bus ID with the adequate Tx Queue. Bits 2:1 are Target Agent ID, bit 3 is the Bus ID. Bits 2:0 are copied into bits 3:1 in the TAG field of the TLP headers of PCIe messages. In CSI platforms, the software device driver programs a value, based on the relevant APIC ID, corresponding to the adequate Tx queue. This value is going to be copied in the 4:0 bits of the *DCA Preferences* field in TLP headers of PCIe messages.
TX Descriptor DCA EN	5	0b	Descriptor DCA EN When set, hardware enables DCA for all Tx descriptors written back into memory. When cleared, hardware does not enable DCA for descriptor write backs. Default cleared. Applies also to head write-back when enabled.
Reserved	7:6	00b	Reserved
TXdescRDNSen	8	0b	Tx Descriptor Read No-Snoop Enable **Note:** This bit must be reset to 0b to ensure correct functionality (except if the software device driver has written this bit with write-through instruction).
TXdescRDROEn	9	1b	Tx Descriptor Read Relax Order Enable
TXdescWBNSen	10	0b	Tx Descriptor Write Back No-Snoop Enable **Note:** This bit must be reset to 0b to ensure correct functionality of descriptor write-back. Applies also to head write-back when enabled.
TXdescWBROEn	11	1b	Tx Descriptor Write Back Relaxed Order Enable Applies also to head write-back when enabled.
TXDataReadNSEn	12	0b	Tx Data Read No-Snoop Enable
TXDataReadROEn	13	1b	Tx Data Read Relax Order Enable
Reserved	31:14	0x0	Reserved

4.4.3.7.11 Transmit IPG Control – TIPG (0x0CB00; RW)

This register controls the Inter Packet Gap (IPG) timer. IPGT specifies the extension to the IPG length for back-to-back transmissions.

Field	Bit(s)	Initial Value	Description
IPGT	7:0	0x0	IPG Transmit Time Measured in increments of 4-byte times. **Note:** For values greater than zero, the 82598 might violate the flow control timing specification (from XOFF packet received to stopping the transmit side).
Reserved	31:8	0x0	Reserved

4.4.3.7.12 Transmit Packet Buffer Size – TXPBSIZE (0x0CC00 – 0x0CC1C; RW)

Field	Bit(s)	Initial Value	Description
Reserved	9:0	0x0	Reserved
SIZE	19:10	0x28/0	Transmit Packet Buffer Size Default values: 0x28 (40 kB) for TXPBSIZE0. 0x0 (0 kB) for RXPBSIZE1-7. Other than the default configuration of one packet buffer, the 82598 supports a partitioned configuration. Partitioned transmit equal: 0x28 (40 kB) for TXPBSIZE0-7.
Reserved	31:20	0x0	Reserved

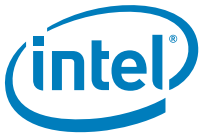
4.4.3.7.13 Manageability Transmit TC Mapping – MNGTXMAP (0x0CD10; RW)

Field	Bit(s)	Initial Value	Description
MAP	2:0	0x0	MAP value indicates the TC that the transmit Manageability traffic is routed to.
Reserved	31:3	0x0	Reserved

4.4.3.8 Wake-Up Control Registers

4.4.3.8.1 Wake Up Control Register – WUC (0x05800; RW)

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved
PME_En	1	0b	PME_En This read/write bit is used by the software device driver to access the PME_En bit of the Power Management Control/Status Register (PMCSR) without writing to PCIe configuration space.
PME_Status (RO)	2	0b	PME_Status This bit is set when the 82598 receives a wake-up 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 it. The PME_Status bit in the PMCSR is also cleared.
Reserved	3	0b	Reserved
ADVD3WUC	4	1b[1]	D3Cold WakeUp Capability Advertisement Enable When set, D3Cold wakeup capability is advertised based on whether the AUX_PWR advertises the presence of auxiliary power (yes if AUX_PWR is indicated, no otherwise). When 0b; however, D3Cold wakeup capability is not advertised even if AUX_PWR presence is indicated. The data value and initial value is EEPROM-configurable.
Reserved	31:5	0b	Reserved



1. Loaded from the EEPROM.

The PME_En and PME_Status bits are reset when Internal Power On Reset or LAN_PWR_GOOD is 0b. When AUX_PWR = 0b or ADVD3WUC=0, these bits are also reset by asserting PE_RST_N.

4.4.3.8.2 Wake Up Filter Control Register – WUFC (0x05808; RW)

Field	Bit(s)	Initial Value	Description
LNKC	0	0b	Link Status Change Wake Up Enable
MAG	1	0b	Magic Packet Wake Up Enable
EX	2	0b	Directed Exact Wake Up Enable
MC	3	0b	Directed Multicast Wake Up Enable
BC	4	0b	Broadcast Wake Up Enable
ARP	5	0b	ARP/IPv4 Request Packet Wake Up Enable
IPV4	6	0b	Directed IPv4 Packet Wake Up Enable
IPV6	7	0b	Directed IPv6 Packet Wake Up Enable
Reserved	14:8	0x0	Reserved
NoTCO	15	0b	Ignore TCO Packets for TCO
FLX0	16	0b	Flexible Filter 0 Enable
Field	**Bit(s)**	**Initial Value**	**Description**
FLX1	17	0b	Flexible Filter 1 Enable
FLX2	18	0b	Flexible Filter 2 Enable
FLX3	19	0b	Flexible Filter 3 Enable
Reserved	31:20	0x0	Reserved

This register is used to enable each of the pre-defined and flexible filters for wake up support. A value of one means the filter is turned on, and a value of zero means the filter is turned off.

If the *NoTCO* bit is set, then any packet that passes the manageability packet filtering does not cause a wake-up event even if it passes one of the wake-up filters.

4.4.3.8.3 Wake Up Status Register – WUS (0x05810; RO)

Field	Bit(s)	Initial Value	Description
LNKC	0	0b	Link Status Changed
MAG	1	0b	Magic Packet Received

EX	2	0b	Directed Exact Packet Received The packet's address matched one of the 16 pre-programmed exact values in the *Receive Address* registers.
MC	3	0b	Directed Multicast Packet Received The packet was a multicast packet whose hashed to a value that corresponded to a 1 bit in the *Multicast Table Array*.
BC	4	0b	Broadcast Packet Received
ARP	5	0b	ARP/IPv4 Request Packet Received
IPV4	6	0b	Directed IPv4 Packet Received
IPV6	7	0b	Directed IPv6 Packet Received
MNG	8	0b	Indicates that a manageability event that should cause a PME to happen.
Reserved	15:9	0x0	Reserved
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
Reserved	31:20	0x0	Reserved

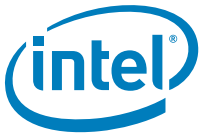
This register is used to record statistics about wake-up packets received. If a packet matches multiple criteria, multiple bits could be set. Writing a 1b to any bit clears that bit.

This register is not cleared when PE_RST_N is asserted. It is only cleared at Internal Power On Reset or LAN_PWR_GOOD, or when cleared by the software device driver.

4.4.3.8.4 IP Address Valid – IPAV (0x5838; RW)

The IP address valid indicates whether the IP addresses in the IP address table are valid.

Field	Bit(s)	Initial Value	Description
V40	0	0	IPv4 Address 0 Valid
V41	1	0	IPv4 Address 1 Valid
V42	2	0	IPv4 Address 2 Valid
V43	3	0	IPv4 Address 3 Valid
Reserved	15:4	0	Reserved
V60	16	0	IPv6 Address 0 Valid
Reserved	31:17	0	Reserved



4.4.3.8.5 IPv4 Address Table – IP4AT (0x05840 + n*8 [n = 0..3]; RW)

The IPv4 address table stores the four IPv4 addresses for ARP/IPv4 request packet and directed IPv4 packet wake up. It has the following format.

DWORD#	Address	31	0
0	0x5840	IPV4ADDR0	
2	0x5848	IPV4ADDR1	
3	0x5850	IPV4ADDR2	
4	0x5858	IPV4ADDR3	

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV4ADDR0	0	0x5840	31:0	X	IPv4 Address 0 (least significant byte is first on the wire).
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.

IPV4ADDR	31:0	X	IPv4 Address

4.4.3.8.6 IPv6 Address Table – IP6AT (0x05880-0x0588C; RW)

The IPv6 address table stores the IPv6 addresses for neighbor discovery packet filtering and directed IPv6 packet wake up and it has the following format.

DWORD#	Address	31 0
0	0x5880	IPV6ADDR0
1	0x5884	
2	0x5888	
3	0x588C	

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV6ADDR0	0	0x5880	31:0	X	IPv6 Address 0, bytes 1-4 (least significant byte is first on the wire).
	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.

Field	Bit(s)	Initial Value	Description
IPV6ADDR	31:0	X	Part of IPv6 address bytes.

4.4.3.8.7 Wake Up Packet Length – WUPL (0x05900; R)

Field	Bit(s)	Initial Value	Description
LEN	15:0	X	Length of wakeup packet.
Reserved	31:16	0x0	Reserved

This register indicates the length of the first wakeup packet received. It is valid if one of the bits in the Wake Up Status (WUS) register is set. It is not cleared by any reset.

4.4.3.8.8 Wake Up Packet Memory (128 Bytes) – WUPM (0x05A00-0x05A7C; R)

Field	Bit(s)	Initial Value	Description
WUPD	31:0	X	Wake Up Packet Data

This register is read only and is used to store the first 128 bytes of the wake up packet for software retrieval after the system wakes. It is not cleared by any reset.

4.4.3.8.9 Flexible Host Filter Table Registers – FHFT (0x09000 – 0x093FC; RW)

Each of the four Flexible Host Filters Table registers (FHFT) 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.

The length field must be eight-byte aligned. For filtering packets shorter than eight-byte aligned, the values should be rounded up to the next eight-byte aligned value. The hardware implementation compares eight bytes at a time so it should get extra zero masks (if needed) until the end of the length value.

If the actual length (defined by the length field register and the mask bits) is not eight-byte aligned, there might be a case in which a packet that is shorter than the actual required length passes the flexible filter. This might happen because of a comparison of up to seven bytes that come after the packet, but that are not really part of the packet.

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 an eight-byte aligned value. If actual packet length is less than (length – 8; length is the value specified by the length field), the filter fails. Otherwise, acceptance depends on the result of actual byte comparison. The value should not be greater than 128.

31 8	31 8	7 0	31 0	31 0
Reserved	Reserved	Mask [7:0]	Dword 1	Dword 0
Reserved	Reserved	Mask [15:8]	Dword 3	Dword 2
Reserved	Reserved	Mask [23:16]	Dword 5	Dword 4
Reserved	Reserved	Mask [31:24]	Dword 7	Dword 6

... ...

31 8	31 8	7 0	31 0	31 0
Reserved	Reserved	Mask [127:120]	Dword 29	Dword 28
Length	Reserved	Mask [127:120]	Dword 31	Dword 30

Field	Dword	Address	Bit(s)	Initial Value
Filter 0 Dword0	0	0x09000	31:0	X
Filter 0 Dword1	1	0x09004	31:0	X
Filter 0 Mask[7:0]	2	0x09008	7:0	X
Reserved	3	0x0900C		X
Filter 0 Dword2	4	0x09010	31:0	X
…				
Filter 0 Dword30	60	0x090F0	31:0	X
Filter 0 Dword31	61	0x090F4	31:0	X
Filter 0 Mask[127:120]	62	0x090F8	7:0	X
Length	63	0x090FC	6:0	X

Accessing the FHFT registers during filter operation might result in a packet being mis-classified if the write operation collides with packet reception. Therefore, flex filters should be disabled prior to changing their setup.

4.4.3.9 Statistic Registers

All statistics registers reset when read. In addition, they stick at 0xFFFF_FFFF when the maximum value is reached.

For the receive statistics, note that a packet is indicated as received if it passes the 82598'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 paths between interrupt-generation and logging of relevant statistics counts, it might be possible to generate an interrupt to the system for an event prior to the associated statistics count actually being incremented. This is 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 reading the statistics might be necessary to avoid the potential for receiving an interrupt and observing an inconsistent statistics count as part of the ISR.

4.4.3.9.1 CRC Error Count – CRCERRS (0x04000; R)

Field	Bit(s)	Initial Value	Description
CRCERRS	31:0	0x0	CRC Error Count

Counts the number of receive packets with CRC errors. In order for a packet to be counted in this register, it must be 64 bytes or greater (from <Destination Address> through <CRC> inclusively) in length. If receives are not enabled, then this register does not increment. This register counts all packets received, not just packets that are directed to the 82598.

4.4.3.9.2 Illegal Byte Error Count – ILLERRC (0x04004; R)

Field	Bit(s)	Initial Value	Description
ILLERRC	31:0	0x0	Illegal Byte Error Count

Counts the number of receive packets with illegal bytes errors (an illegal symbol in the packet). This register counts all packets received, not just packets that are directed to the 82598.

4.4.3.9.3 Error Byte Count – ERRBC (0x04008; R)

Field	Bit(s)	Initial Value	Description
ERRBC	31:0	0x0	Error Byte Count

Counts the number of receive packets with Error bytes (an error symbol in the packet). This register counts all packets received, not just packets that are directed to the 82598.

4.4.3.9.4 MAC Short Packet Discard Count – MSPDC (0x04010; R)

Field	Bit(s)	Initial Value	Description
MSPDC	31:0	0x0	Number of MAC short packet discard packets received.

4.4.3.9.5 Missed Packets Count – MPC (0x03FA0 – 0x03FBC; R)

Field	Bit(s)	Initial Value	Description
MPC	31:0	0x0	Missed Packets Count

This counter counts the number of missed packets per packet buffer. Packets are missed when the receive FIFO has insufficient space to store the incoming packet. This could be caused because of too few buffers allocated, or because there is insufficient bandwidth on the IO bus. This register does not increment if receives are not enabled.

4.4.3.9.6 MAC Local Fault Count – MLFC (0x04034; R)

Field	Bit(s)	Initial Value	Description
MLFC	31:0	0x0	Number of faults in the local MAC. **Note:** For proper counting this statistics should be cleared after link up. **Note:** This statistics field is only valid when the link speed is 10 Gb/s.

4.4.3.9.7 MAC Remote Fault Count – MRFC (0x04038; R)

Field	Bit(s)	Initial Value	Description
MRFC	31:0	0x0	Number of faults in the remote MAC. **Note:** For proper counting this statistics should be cleared after link up. Note: This statistics field is only valid when the link speed is 10 Gb/s.

4.4.3.9.8 Receive Length Error Count – RLEC (0x04040; R)

Field	Bit(s)	Initial Value	Description
RLEC	31:0	0x0	Number of packets with receive length errors.

This register counts receive length error events. A length error occurs if an incoming packet length field in the MAC header doesn't match the packet length. To enable the receive length error count HLREG.RXLNGTHERREN bit needs to be set to 1b.

4.4.3.9.9 Link XON Transmitted Count – LXONTXC (0x03F60; R)

Field	Bit(s)	Initial Value	Description
LXONTXC	31:0	0x0	Number of XON packets transmitted.

This register counts the number of XON packets received per user priority. XON packets can use the global address, or the station address.

4.4.3.9.10 Link XON Received Count – LXONRXC (0x0CF60; R)

Field	Bit(s)	Initial Value	Description
LXONRXC	31:0	0x0	Number of XON packets received.

This register counts the number of XON packets transmitted per user priority. These can be either due to queue fullness, or due to software initiated action (using SWXOFF).

4.4.3.9.11 Link XOFF Transmitted Count – LXOFFTXC (0x03F68; R)

Field	Bit(s)	Initial Value	Description
LXOFFTXC	31:0	0x0	Number of XOFF packets transmitted.

This register counts the number of XOFF packets received per user priority. XOFF packets can use the global address, or the station address.

4.4.3.9.12 Link XOFF Received Count – LXOFFRXC (0x0CF68; R)

Field	Bit(s)	Initial Value	Description
LXOFFRXC	31:0	0x0	Number of XOFF packets received.

This register counts the number of XOFF packets transmitted per user priority. These can be either due to queue fullness, or due to software initiated action (using SWXOFF).

4.4.3.9.13 Priority XON Transmitted Count – PXONTXC (0x03F00 – 0x03F1C; R)

Field	Bit(s)	Initial Value	Description
PXONTXC	31:0	0x0	Number of XON packets transmitted.

This register counts the number of XON packets received per user priority. XON packets can use the global address, or the station address.

4.4.3.9.14 Priority XON Received Count – PXONRXC (0x0CF00 – 0x0CF1C; R)

Field	Bit(s)	Initial Value	Description
PXONRXC	31:0	0x0	Number of XON packets received

This register counts the number of XON packets transmitted per user priority. These can be either due to queue fullness, or due to software initiated action (using SWXOFF).

4.4.3.9.15 Priority XOFF Transmitted Count – PXOFFTXC (0x03F20 – 0x03F3C; R)

Field	Bit(s)	Initial Value	Description
PXOFFTXC	31:0	0x0	Number of XOFF packets transmitted.

This register counts the number of XOFF packets received per user priority. XOFF packets can use the global address, or the station address.

4.4.3.9.16 Priority XOFF Received Count – PXOFFRXC (0x0CF20 – 0x0CF2C; R)

Field	Bit(s)	Initial Value	Description
PXOFFRXC	31:0	0x0	Number of XOFF packets received.

This register counts the number of XOFF packets transmitted per user priority. These can be either due to queue fullness, or due to software initiated action (using SWXOFF).

4.4.3.9.17 Packets Received (64 Bytes) Count – PRC64 (0x0405C; R)

Field	Bit(s)	Initial Value	Description
PRC64	31:0	0x0	Number of packets received that are 64 bytes in length.

This register counts the number of good packets received that are exactly 64 bytes (from <Destination Address> through <CRC>, inclusively) 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.

4.4.3.9.18 Packets Received (65-127 Bytes) Count – PRC127 (0x04060; R)

Field	Bit(s)	Initial Value	Description
PRC127	31:0	0	Number of packets received that are 65-127 bytes in length.

This register counts the number of good packets received that are 65-127 bytes (from <Destination Address> through <CRC>, inclusively) 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.

4.4.3.9.19 Packets Received (128-255 Bytes) Count – PRC255 (0x04064; R)

Field	Bit(s)	Initial Value	Description
PRC255	31:0	0x0	Number of packets received that are 128-255 bytes in length.

This register counts the number of good packets received that are 128-255 bytes (from <Destination Address> through <CRC>, inclusively) 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.

4.4.3.9.20 Packets Received (256-511 Bytes) Count – PRC511 (0x04068; R)

Field	Bit(s)	Initial Value	Description
PRC511	31:0	0x0	Number of packets received that are 256-511 bytes in length.

This register counts the number of good packets received that are 256-511 bytes (from <Destination Address> through <CRC>, inclusively) 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.

4.4.3.9.21 Packets Received (512-1023 Bytes) Count – PRC1023 (0x0406C; R)

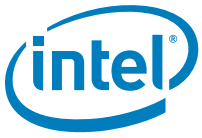
Field	Bit(s)	Initial Value	Description
PRC1023	31:0	0x0	Number of packets received that are 512-1023 bytes in length.

This register counts the number of good packets received that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusively) 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.

4.4.3.9.22 Packets Received (1024 to Max Bytes) Count – PRC1522 (0x04070; R)

Field	Bit(s)	Initial Value	Description
PRC1522	31:0	0x0	Number of packets received that are 1024-Max bytes in length.

This register counts the number of good packets received that are from 1024 bytes to the maximum (from <Destination Address> through <CRC>, inclusively) in length. The maximum is dependent on the current receiver configuration and the type of packet being received. If a packet is counted in Receive Oversized Count, it is not counted in this register (see Section 4.4.3.9.32). This register does not include received flow control packets and only increments if the packet has passed address filtering and receives are enabled.



Due to changes in the standard for maximum frame size for VLAN tagged frames in 802.3, the 82598 accepts packets that have a maximum length of 1522 bytes. RMON statistics associated with this range has been extended to count 1522 byte long packets.

4.4.3.9.23 Good Packets Received Count – GPRC (0x04074; R)

Field	Bit(s)	Initial Value	Description
GPRC	31:0	0x0	Number of good packets received (of any length).

This register counts the number of good (non-erred) packets received of any legal length. The legal length for the received packet is defined by the value of LongPacketEnable (see Section 4.4.3.9.8). The register does not include received flow control packets and only counts packets that pass filtering. It only increments if receives are enabled and does not tally packets counted by the Missed Packet Count (MPC) register.

GPRC might count packets interrupted by link disconnect although they have a CRC error

4.4.3.9.24 Broadcast Packets Received Count – BPRC (0x04078; R)

Field	Bit(s)	Initial Value	Description
BPRC	31:0	0x0	Number of broadcast packets received.

This register counts the number of good (non-erred) broadcast packets received. It does not count broadcast packets received when the broadcast address filter is disabled and only increments if receives are enabled.

4.4.3.9.25 Multicast Packets Received Count – MPRC (0x0407C; R)

Field	Bit(s)	Initial Value	Description
MPRC	31:0	0x0	Number of multicast packets received.

This register counts the number of good (non-erred) multicast packets received. It does not tally multicast packets received that fail to pass address filtering or received flow control packets. This register only increments if receives are enabled and does not tally packets counted by the Missed Packet Count (MPC) register.

4.4.3.9.26 Good Packets Transmitted Count – GPTC (0x04080; R)

Field	Bit(s)	Initial Value	Description
GPTC	31:0	0x0	Number of good packets transmitted.

This register counts the number of good (non-erred) packets transmitted, including flow control packets. A good transmit packet is one that is 64 or more bytes in length (from <Destination Address> through <CRC>, inclusively) in length. The register only increments if transmits are enabled and does not count packets counted by the Missed Packet Count (MPC) register. The register counts clear as well as secure packets.

4.4.3.9.27 Good Octets Received Count – GORC (0x0408C; R)

Field	Bit(s)	Initial Value	Description
GORC	31:0	0x0	Number of good octets received.

This register counts the number of good (non-erred) octets received, including flow control packets. It includes bytes received in a packet from the *Destination Address* field through the *CRC* field, inclusively.

In addition, it sticks at 0xFFFF_FFFF when the maximum value is reached. Only packets that pass address filtering are counted in this register and it only increments if receives are enabled.

4.4.3.9.28 Good Octets Transmitted Count – GOTC (0x04094; R);

Field	Bit(s)	Initial Value	Description
GOTC	31:0	0x0	Number of good octets transmitted.

This register counts the number of good (non-erred) packets transmitted, including flow control packets.

In addition, it sticks at 0xFFFF_FFFF when the maximum value is reached. This register includes bytes transmitted in a packet from the *Destination Address* field through the *CRC* field, inclusively. It counts octets in successfully transmitted packets and only increments if transmits are enabled. It also counts clear as well as secure octets.

4.4.3.9.29 Receive No Buffers Count – RNBC (0x03FC0 – 0x03FDC; R)

Field	Bit(s)	Initial Value	Description
RNBC	31:0	0x0	Number of receive no buffer conditions.

This register counts the number of times frames were received when there were no available buffers in the appropriate queue to store the frames or the queue was disabled. The packet is still received if there is space in the FIFO and the Drop_En bit for the target queue is clear (0b).

This register only increments if receives are enabled and does not increment when flow control packets are received.

4.4.3.9.30 Receive Undersize Count – RUC (0x040A4; R)

Field	Bit(s)	Initial Value	Description
RUC	31:0	0x0	Number of receive undersize errors.

This register counts the number of received frames that passed address filtering, were less than minimum size (64 bytes from <Destination Address> through <CRC>, inclusively), and had a valid CRC. It only increments if receives are enabled.

4.4.3.9.31 Receive Fragment Count – RFC (0x040A8; R)

Field	Bit(s)	Initial Value	Description
RFC	31:0	0x0	Number of receive fragment errors.

This register counts the number of received frames that pass address filtering, are less than minimum size (64 bytes from <Destination Address> through <CRC>, inclusively), and have a bad CRC. This is slightly different from the Receive Undersize Count register. The register only increments if receives are enabled.

4.4.3.9.32 Receive Oversize Count – ROC (0x040AC; R)

Field	Bit(s)	Initial Value	Description
ROC	31:0	0x0	Number of receive oversize errors.

This register counts the number of received frames that pass address filtering and are greater than maximum size. An oversized packet is defined according to MHADD.MFS. See Section 4.4.3.9.21.

If receives are not enabled, the register does not increment. Lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusively.

4.4.3.9.33 Receive Jabber Count – RJC (0x040B0; R)

Field	Bit(s)	Initial Value	Description
RJC	31:0	0x0	Number of receive jabber errors.

This register counts the number of received frames that pass address filtering, are were greater than maximum size and have a bad CRC. This is slightly different from the Receive Oversize Count register.

If receives are not enabled, the register does not increment. These lengths include bytes in the received packet from <Destination Address> through <CRC>, inclusively.

4.4.3.9.34 Management Packets Received Count – MNGPRC (0x040B4; R)

This register counts the total number of packets received that pass management filters. Management packets include RMCP and ARP packets. Packets with errors are not counted; packets dropped because the management receive FIFO is full are counted.

Field	Bit(s)	Initial Value	Description
MNGPRC	31:0	0	Number of management packets received.

4.4.3.9.35 Management Packets Dropped Count – MNGPDC (0x040B8; R)

This register counts the total number of packets received that pass the management filters and then are dropped because the management receive FIFO is full. Management packets include any packet directed to the manageability console, such as RMCP and ARP packets.

Field	Bit(s)	Initial Value	Description
MNGPDC	31:0	0x0	Number of management packets dropped.

4.4.3.9.36 Management Packets Transmitted Count – MNGPTC (0x0CF90; R)

This register counts the total number of packets that are transmitted or received over the SMBus.

Field	Bit(s)	Initial Value	Description
MNGPTC	31:0	0x0	Number of management packets transmitted.

4.4.3.9.37 Total Octets Received – TOR (0x040C4; R);

Field	Bit(s)	Initial Value	Description
TOR	31:0	0x0	Number of total octets received – upper 4 bytes.

This register counts the total number of octets received. In addition, it sticks at 0xFFFF_FFFF when the maximum value is reached.

All packets received passing at least one of the L2 receive filters have their octets summed into this register, regardless of their length, whether they are erred, or whether they are flow control packets. It includes bytes received in a packet from the *Destination Address* field through the *CRC* field, inclusively. This register only increments if receives are enabled.

Broadcast rejected packets are counted in this counter (in contradiction to all other rejected packets that are not counted).

4.4.3.9.38 Total Packets Received – TPR (0x040D0; R)

Field	Bit(s)	Initial Value	Description
TPR	31:0	0x0	Number of all packets received.

This register counts the total number of all packets received. All packets received are counted regardless of their length, whether they are erred, or whether they are flow control packets. The register only increments if receives are enabled.

Broadcast rejected packets are counted in this counter (in contradiction to all other rejected packets that are not counted). TPR might count packets interrupted by link disconnect although they have a CRC error.

4.4.3.9.39 Total Packets Transmitted – TPT (0x040D4; R)

Field	Bit(s)	Initial Value	Description
TPT	31:0	0x0	Number of all packets transmitted.

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 tallied. This register only increments if transmits are enabled. It counts all packets, including standard packets, secure packets, packets received over the SMBus.

4.4.3.9.40 Packets Transmitted (64 Bytes) Count – PTC64 (0x040D8; R)

Field	Bit(s)	Initial Value	Description
PTC64	31:0	0x0	Number of packets transmitted that are 64 bytes in length.

This register counts the number of packets transmitted that are exactly 64 bytes (from <Destination Address> through <CRC>, inclusively) in length, including flow control packets. Partial packet transmissions (collisions in half-duplex mode) are not tallied. It only increments if transmits are enabled and counts all other packets, including: standard packets, secure packets, packets received over the SMBus.

4.4.3.9.41 Packets Transmitted (65-127 Bytes) Count – PTC127 (0x040DC; R)

Field	Bit(s)	Initial Value	Description
PTC127	31:0	0x0	Number of packets transmitted that are 65-127 bytes in length.

This register counts the number of packets transmitted that are 65-127 bytes (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (collisions in half-duplex mode) are not tallied. This register only increments if transmits are enabled. This register counts all packets, including: standard packets, secure packets, packets received over the SMBus.

4.4.3.9.42 Packets Transmitted (128-255 Bytes) Count – PTC255 (0x040E0; R)

Field	Bit(s)	Initial Value	Description
PTC255	31:0	0x0	Number of packets transmitted that are 128-255 bytes in length.

This register counts the number of packets transmitted that are 128-255 bytes (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (collisions in half-duplex mode) are not tallied. This register only increments if transmits are enabled and counts all packets, including: standard packets, secure packets, packets received over the SMBus.

4.4.3.9.43 Packets Transmitted (256-511 Bytes) Count – PTC511 (0x040E4; R)

Field	Bit(s)	Initial Value	Description
PTC511	31:0	0x0	Number of packets transmitted that are 256-511 bytes in length.

This register counts the number of packets transmitted that are 256-511 bytes (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled and counts all packets, including: standard and secure packets (management packets are never be more than 200 bytes).

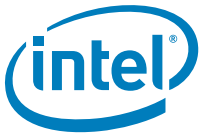
4.4.3.9.44 Packets Transmitted (512-1023 Bytes) Count – PTC1023 (0x040E8; R)

Field	Bit(s)	Initial Value	Description
PTC1023	31:0	0x0	Number of packets transmitted that are 512-1023 bytes in length.

This register counts the number of packets transmitted that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusively) in length. Partial packet transmissions (collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled and counts all packets, including: standard and secure packets (management packets are never be more than 200 bytes).

4.4.3.9.45 Packets Transmitted (Greater than 1024 Bytes) Count – PTC1522 (0x040EC; R)

Field	Bit(s)	Initial Value	Description
PTC1522	31:0	0x0	Number of packets transmitted that are 1024 or more bytes in length.



This register counts the number of packets transmitted that are 1024 or more bytes (from <Destination Address> through <CRC>, inclusively) in length. This register only increments if transmits are enabled.

Due to changes in the standard for maximum frame size for VLAN tagged frames in 802.3, this device transmits packets which have a maximum length of 1522 bytes. RMON statistics associated with this range has been extended to count 1522 byte long packets. This register counts all packets, including standard and secure packets (management packets are never be more than 200 bytes).

4.4.3.9.46 Multicast Packets Transmitted Count – MPTC (0x040F0; R)

Field	Bit(s)	Initial Value	Description
MPTC	31:0	0x0	Number of multicast packets transmitted.

This register counts the number of multicast packets transmitted, including flow control packets. Counts clear as well as secure traffic.

4.4.3.9.47 Broadcast Packets Transmitted Count – BPTC (0x040F4; R)

Field	Bit(s)	Initial Value	Description
BPTC	31:0	0x0	Number of broadcast packets transmitted count.

This register counts the number of broadcast packets transmitted. It only increments if transmits are enabled and counts all packets, including standard and secure packets (management packets are never be more than 200 bytes).

After a broadcast packet is sent by the host, all flow control and manageability packets that are sent are counted as Broadcast packets until a non-broadcast packet is sent by the host.

4.4.3.9.48 XSUM Error Count – XEC (0x04120; RO)

Field	Bit(s)	Initial Value	Description
XEC	31:0	0x0	Number of receive IPv4, TCP, UDP checksum errors

XSUM errors are not counted when a packet has MAC error (CRC, length, under-size, over-size, byte error or symbol error).

4.4.3.9.49 Receive Queue Statistic Mapping Registers RQSMR (0x2300 + 4*n [n=0…15], RW)

These registers define the mapping of the receive queues to the per-queue statistics. This mapping maps the queues to statistic registers QPRC and QBRC (note that there are 16 of each).

There are 64 queues and only 16 queue statistics registers so each entry refers to a queue and the value indicates which QPRC and QBRC of the 16 this queue statistics is being counted.

Several queues can be mapped to a single statistic register. Each statistic register counts the number of packets and bytes of all queues that are mapped to that statistics.

31 24	23 16	15 8	7 0
Q_MAP[3]	Q_MAP[2]	Q_MAP[1]	Q_MAP[0]
...	...	...	...

... ...

...	...	...	...
Q_MAP[63]	Q_MAP[62]	Q_MAP[61]	Q_MAP[60]

Field	Bit(s)	Initial Value	Description
Q_MAP[0]	3:0	0x0	Defines the per-queue statistic register that is mapped to this queue.
Reserved	7:4	0x0	Reserved
Q_MAP[1]	11:8	0x0	Defines the per-queue statistic register that is mapped to this queue.
Reserved	15:12	0x0	Reserved
Q_MAP[2]	19:16	0x0	Defines the per-queue statistic register that is mapped to this queue.
Reserved	23:20	0x0	Reserved
Q_MAP[3]	27:24	0x0	Defines the per-queue statistic register that is mapped to this queue.
Reserved	31:28	0x0	Reserved

4.4.3.9.50 Transmit Queue Statistic Mapping Registers TQSMR (0x7300 + 4*n [n=0...7], RW)

These registers define the mapping of the transmit queues to the per-queue statistics. This mapping maps the queues to statistic registers QPTC and QBTC (note that there are 16 of each).

There are 64 queues and only 16 queue statistics registers so each entry refers to a queue and the value indicates which QPTC and QBTC of the 16 this queue statistics is being counted.

Several queues can be mapped to a single statistic register. Each statistic register counts the number of packets and bytes of all queues that are mapped to that statistics.

31 24	23 16	15 8	7 0
Q_MAP[3]	Q_MAP[2]	Q_MAP[1]	Q_MAP[0]
...	...	...	...

. . .

...	...	...	...
Q_MAP[31]	Q_MAP[30]	Q_MAP[29]	Q_MAP[28]

Field	Bit(s)	Initial Value	Description
Q_MAP[0]	3:0	0x0	Defines the per-queue statistic register that is mapped to this queue.
Reserved	7:4	0x0	Reserved
Q_MAP[1]	11:8	0x0	Defines the per-queue statistic register that is mapped to this queue.
Reserved	15:12	0x0	Reserved
Q_MAP[2]	19:16	0x0	Defines the per-queue statistic register that is mapped to this queue.
Reserved	23:20	0x0	Reserved
Q_MAP[3]	27:24	0x0	Defines the per-queue statistic register that is mapped to this queue.
Reserved	31:28	0x0	Reserved

4.4.3.9.51 Queue Packets Received Count – QPRC (0x01030+ n*0x40[n=0..15]; R)

Field	Bit(s)	Initial Value	Description
QPRC	31:0	0x0	Number of packets received for the queue.

4.4.3.9.52 Queue Packets Transmitted Count – QPTC (0x06030 + n*0x40[n=0..15]; R)

Field	Bit(s)	Initial Value	Description
QPTC	31:0	0x0	Number of packets transmitted for the queue.

4.4.3.9.53 Queue Bytes Received Count – QBRC (0x1034 + n*0x40[n=0..15]; R)

Field	Bit(s)	Initial Value	Description
QBRC	31:0	0x0	Number of bytes received for the queue.

4.4.3.9.54 Queue Bytes Transmitted Count – QBTC (0x6034+n*0x40[n=0..15]; R)

Field	Bit(s)	Initial Value	Description
QBTC	31:0	0x0	Number of bytes transmitted for the queue.

4.4.3.10 Management Filter Registers

4.4.3.10.1 Management VLAN TAG Value – MAVTV (0x5010 +4*n[n=0..7]; RW)

Where "n" is the VLAN filter serial number, equal to 0, 1,…7.

MAVTV registers are written by the BMC and are not accessible to the host for writing. The registers are used to filter manageability packets.

Field	Bit(s)	Initial Value	Description
VID	11:0	0x0	Contains the VLAN ID that should be compared with the incoming packet if bit 31 is set.
Reserved	31:12	0x0	Reserved

4.4.3.10.2 Management Flex UDP/TCP Ports – MFUTP (0x5030 + 4*n[n=0..7]; RW)

Where each 32-bit register (n=0,…,7) refers to two port filters (register 0 refers to ports 0 and 1, register 1 refers to port 2 and 3, etc).

MFUTP registers are written by the BMC and not accessible to the host for writing.

Reset – MFUTP registers are cleared on Internal Power On Reset or LAN_PWR_GOOD only. The initial values for this register can be loaded from the EEPROM by the management firmware after power-up reset.

MFUTP registers value should be configured to the register in host order.

Field	Bit(s)	Initial Value	Description
MFUTP[2n]	15:0	0x0	(2n)-th management flex UDP/TCP port.
MFUTP[2n+1]	31:16	0x0	(2n+1)-th management flex UDP/TCP port.

4.4.3.10.3 Management Control Register – MANC (0x05820; RW)

Field	Bit(s)	Initial Value	Description
Reserved	16:0	0x0	Reserved

RCV_TCO_EN	17	0b	Receive TCO Packets Enabled When this bit is set, it enables the receive flow from the wire to the manageability block.
Reserved	18	0b	Reserved
RCV_ALL	19	0b	Receive All Enable When set, all packets are received from the wire and passed to the manageability block.
MCST_PASS_L2	20	0b	Multicast Promiscuous When set, all multicast filters pass L2 address filtering (same as the host promiscuous multicast.
EN_MNG2HOST	21	0b	Enable Manageability Packets to Host Memory This bit enables the functionality of the MANC2H register. When set, the packets that are specified in the MANC2H registers are also forwarded to the host memory, if they pass manageability filters.
Reserved	22	0b	Reserved
EN_XSUM_FILTER	23	0b	Enable Checksum Filtering to Manageability When set, only packets that pass L3 and L4 checksums are sent to the manageability block.
EN_IPv4_FILTER	24	0b	Enable IPv4 address Filters When set, the last 128 bits of the MIPAF register are used to store four IPv4 addresses for IPv4 filtering. When cleared, these bits store a single IPv6 filter.
FIXED_NET_TYPE	25	0b	Fixed Next Type Enable If set, only packets matching the net type defined by the NET_TYPE field (bit 26 in this register) passes to manageability.
NET_TYPE	26	0b	Net Type 0b = Pass only un-tagged packets. 1b = Pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE (bit 25) is set. Packet has to pass one MDEF/ RCV_ALL in order to be checked by this rule.
Reserved	31:27	0x0	Reserved

4.4.3.10.4 Manageability Filters Valid – MFVAL (0x5824; RW)

The manageability filters valid registers indicate which filter registers contain a valid entry.

Reset – The MFVAL register is cleared on Internal Power On Reset or LAN_PWR_GOOD reset.

Field	Bit(s)	Initial Value	Description
MAC	3:0	0x0[1]	MAC Indicates if the MAC unicast filter registers (MMAH and MMAL) contain valid MAC addresses. Bit 0 corresponds to filter 0, etc.
Reserved	7:4	0x0[1]	Reserved
VLAN	15:8	0x0[1]	VLAN Indicates if the VLAN filter registers (MAVTV) contain valid VLAN tags. Bit 8 corresponds to filter 0, etc.

IPv4	19:16	0x0[1]	IPv4 Indicates if the IPv4 address filters (MIPAF) contain valid IPv4 addresses. Bit 16 corresponds to IPv4 address 0. These bits apply only when IPv4 address filters are enabled (MANC.EN_IPv4_FILTER=1b)
Reserved	23:20	0x0[1]	Reserved
IPv6	27:24	0x0[1]	IPv6 Indicates if the IPv6 address filter registers (MIPAF) contain valid IPv6 addresses. Bit 24 corresponds to address 0, etc. Bit 27 (filter 3) applies only when IPv4 address filters are not enabled (MANC.EN_IPv4_FILTER=0b).
Reserved	31:28	0x0[1]	Reserved

1. The initial values for this register can be loaded from the EEPROM by the management firmware after power-up reset or firmware reset. The MFVAL register is written by the BMC and not accessible to the host for writing.

4.4.3.10.5 Management Control To Host Register – MANC2H (0x5860; RW)

The MANC2H register enables routing of manageability packets to the host based on the decision filter that routed the packet to the manageability micro-controller. Each manageability decision filter (MDEF) has a corresponding bit in the MANC2H register. When a manageability decision filter (MDEF) routes a packet to manageability, it also routes the packet to the host if the corresponding MANC2HOST bit is set and if the EN_MNG2HOST bit is set. The EN_MNG2HOST bit serves as a global enable for the MANC2H bits.

Reset – The MANC2H register is cleared on Internal Power On Reset or LAN_PWR_GOOD, and firmware reset.

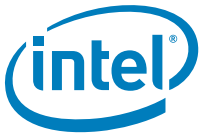
Field	Bit(s)	Initial Value	Description
Host Enable	7:0	0x0[1]	Host Enable When set, indicates that packets routed by the manageability filters to manageability are also sent to the host. Bit 0 corresponds to decision rule 0, etc.
Reserved	31:8	0x0	Reserved

1. The initial values for this register can be loaded from the EEPROM by the management firmware after power-up reset or firmware reset.

4.4.3.10.6 Manageability Decision Filters- MDEF (0x5890 + 4*n[n=0..7]; RW)

Reset – The MDEF registers are cleared on Internal Power On Reset or LAN_PWR_GOOD reset.

Field	Bit(s)	Initial Value	Description
Unicast AND	0	0b[1]	Unicast Controls the inclusion of unicast address filtering in the manageability filter decision (AND section).
Broadcast AND	1	0b[1]	Broadcast Controls the inclusion of broadcast address filtering in the manageability filter decision (AND section).



VLAN AND	2	0b[1]	VLAN Controls the inclusion of VLAN address filtering in the manageability filter decision (AND section).
IP Address	3	0b[1]	IP Address Controls the inclusion of IP address filtering in the manageability filter decision (AND section).
Unicast OR	4	0b[1]	Unicast Controls the inclusion of unicast address filtering in the manageability filter decision (OR section).
Broadcast OR	5	0b[1]	Broadcast Controls the inclusion of broadcast address filtering in the manageability filter decision (OR section).
Multicast AND	6	0b[1]	Multicast Controls the inclusion of multicast address filtering in the manageability filter decision (AND section). Broadcast packets are not included by this bit. The packet must pass some L2 filtering to be included by this bit – either by the MANC.MCST_PASS_L2 or by some dedicated MAC address.
ARP Request	7	0b[1]	ARP Request Controls the inclusion of ARP request filtering in the manageability filter decision (OR section).
ARP Response	8	0b[1]	ARP Response Controls the inclusion of ARP response filtering in the manageability filter decision (OR section).
Reserved	9	0b[1]	Reserved
Port 0x298	10	0b[1]	Port 0x298 Controls the inclusion of port 0x298 filtering in the manageability filter decision (OR section).
Port 0x26F	11	0b[1]	Port 0x26F Controls the inclusion of port 0x26F filtering in the manageability filter decision (OR section).
Field	**Bit(s)**	**Initial Value**	**Description**
Flex port	27:12	0x0[1]	Flex port Controls the inclusion of flex port filtering in the manageability filter decision (OR section). Bit 12 corresponds to flex port 0, etc.
Flex TCO	31:28	0x0[1]	Flex TCO Controls the inclusion of Flex TCO filtering in the manageability filter decision (OR section). Bit 28 corresponds to Flex TCO filter 0, etc.

1. The initial values for this register can be loaded from the EEPROM by the management firmware after power-up reset or firmware reset.

4.4.3.10.7 Manageability IP Address Filter – MIPAF (0x58B0-0x58EC; RW)

The Manageability IP Address Filter register stores IP addresses for manageability filtering. The MIPAF register can be used in two configurations, depending on the value of the MANC. EN_IPv4_FILTER bit:

- EN_IPv4_FILTER = 0b: the last 128 bits of the register store a single IPv6 address (IPV6ADDR3)
- EN_IPv4_FILTER = 1bs: the last 128 bits of the register store four IPv4 addresses (IPV4ADDR[3:0])

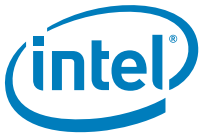
Reset – These registers are cleared on Internal Power On Reset or LAN_PWR_GOOD only.

MIPAF registers value should be configured to the register in host order.

EN_IPv4_FILTER = 0b:

DWORD#	Address	31	0
0	0x58B0	IPV6ADDR0	
1	0x58B4		
2	0x58B8		
3	0x58BC		
4	0x58C0	IPV6ADDR1	
5	0x58C4		
6	0x58C8		
7	0x58CC		
8	0x58D0	IPV6ADDR2	
9	0x58D4		
10	0x58D8		
11	0x58DC		
12	0x58E0	IPV6ADDR3	
13	0x58E4		
14	0x58E8		
15	0x58EC		

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV6ADDR0	0	0x58B0	31:0	X[1]	IPv6 Address 0, bytes 1-4 (least significant byte is first on the wire)
	1	0x58B4	31:0	X[1]	IPv6 Address 0, bytes 5-8
	2	0x58B8	31:0	X[1]	IPv6 Address 0, bytes 9-12



	3	0x58BC	31:0	X[1]	IPv6 Address 0, bytes 16-13
IPV6ADDR1	0	0x58C0	31:0	X[1]	IPv6 Address 1, bytes 1-4 (least significant byte is first on the wire)
	1	0x58C4	31:0	X[1]	IPv6 Address 1, bytes 5-8
	2	0x58C8	31:0	X[1]	IPv6 Address 1, bytes 9-12
	3	0x58CC	31:0	X[1]	IPv6 Address 1, bytes 16-13
IPV6ADDR2	0	0x58D0	31:0	X[1]	IPv6 Address 2, bytes 1-4 (least significant byte is first on the wire)
	1	0x58D4	31:0	X[1]	IPv6 Address 2, bytes 5-8
	2	0x58D8	31:0	X[1]	IPv6 Address 2, bytes 9-12
	3	0x58DC	31:0	X[1]	IPv6 Address 2, bytes 16-13
IPV6ADDR3	0	0x58E0	31:0	X[1]	IPv6 Address 3, bytes 1-4 (least significant byte is first on the wire)
	1	0x58E4	31:0	X[1]	IPv6 Address 3, bytes 5-8
	2	0x58E8	31:0	X[1]	IPv6 Address 3, bytes 9-12
	3	0x58EC	31:0	X[1]	IPv6 Address 3, bytes 16-13

1. The initial values for these registers can be loaded from the EEPROM after power-up reset. The registers are written by the BMC and not accessible to the host for writing.

EN_IPv4_FILTER = 1b:

DWORD#	Address	31	0
0	0x58B0	IPV6ADDR0	
1	0x58B4		
2	0x58B8		
3	0x58BC		
4	0x58C0	IPV6ADDR1	
5	0x58C4		
6	0x58C8		
7	0x58CC		
8	0x58D0	IPV6ADDR2	
9	0x58D4		
10	0x58D8		
11	0x58DC		
12	0x58E0	IPV4ADDR0	

13	0x58E4	IPV4ADDR1
14	0x58E8	IPV4ADDR2
15	0x58EC	IPV4ADDR3

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV6ADDR0	0	0x58B0	31:0	X[1]	IPv6 Address 0, bytes 1-4 (least significant byte is first on the wire)
	1	0x58B4	31:0	X[1]	IPv6 Address 0, bytes 5-8
	2	0x58B8	31:0	X[1]	IPv6 Address 0, bytes 9-12
	3	0x58BC	31:0	X[1]	IPv6 Address 0, bytes 16-13
IPV6ADDR1	0	0x58C0	31:0	X[1]	IPv6 Address 1, bytes 1-4 (least significant byte is first on the wire)
	1	0x58C4	31:0	X[1]	IPv6 Address 1, bytes 5-8
	2	0x58C8	31:0	X[1]	IPv6 Address 1, bytes 9-12
	3	0x58CC	31:0	X[1]	IPv6 Address 1, bytes 16-13
IPV6ADDR2	0	0x58D0	31:0	X[1]	IPv6 Address 2, bytes 1-4 (least significant byte is first on the wire)
	1	0x58D4	31:0	X[1]	IPv6 Address 2, bytes 5-8
	2	0x58D8	31:0	X[1]	IPv6 Address 2, bytes 9-12
	3	0x58DC	31:0	X[1]	IPv6 Address 2, bytes 16-13
IPV4ADDR0	0	0x58E0	31:0	X[1]	IPv4 Address 0 (least significant byte is first on the wire)
IPV4ADDR1	1	0x58E4	31:0	X[1]	IPv4 Address 1 (least significant byte is first on the wire)
IPV4ADDR2	2	0x58E8	31:0	X[1]	IPv4 Address 2 (least significant byte is first on the wire)
IPV4ADDR3	3	0x58EC	31:0	X[1]	IPv4 Address 3 (least significant byte is first on the wire)

1. The initial values for these registers can be loaded from the EEPROM after power-up reset. The registers are written by the BMC and not accessible to the host for writing.

Field	Bit(s)	Initial Value	Description
IP_ADDR 4 bytes	31:0	X[1]	Four bytes of IP (v6 or v4) address i mod 4 = 0 to bytes 1 – 4 i mod 4 = 1 to bytes 5 – 8 i mod 4 = 0 to bytes 9 – 12 i mod 4 = 0 to bytes 13 – 16 where i div four is the index of IP address (0..3).

1. The initial values for these registers can be loaded from the EEPROM after power-up reset. The registers are written by the BMC and not accessible to the host for writing.

4.4.3.10.8 Manageability MAC Address Low – MMAL (0x5910 + 8*n[n=0..3]; RW)

These registers contain the lower bits of the 48-bit Ethernet address. MMAL registers are written by the BMC and not accessible to the host for writing. They are used to filter manageability packets.

Reset – MMAL registers are cleared on Internal Power On Reset or LAN_PWR_GOOD only.

The MMAL value should be configured to the register in host order.

Field	Bit(s)	Initial Value	Description
MMAL	31:0	X[1]	Manageability MAC Address Low The lower 32 bits of the 48-bit Ethernet address.

1. The initial values for this register can be loaded from the EEPROM by the management firmware after power-up reset.

4.4.3.10.9 Manageability MAC Address High – MMAH (0x5914 + 8*n[n=0..3]; RW)

These registers contain the upper bits of the 48-bit Ethernet address. The complete address is {MMAH, MMAL}. MMAH registers are written by the BMC and not accessible to the host for writing. They are used to filter manageability packets.

Reset – MMAL registers are cleared on Internal Power On Reset or LAN_PWR_GOOD only.

The MMAH value should be configured to the register in host order.

Field	Bit(s)	Initial Value	Description
MMAH	15:0	X[1]	Manageability MAC Address High The upper 16 bits of the 48-bit Ethernet address.
Reserved	31:16	0x0	Reserved Reads as 0x0. Ignored on writes.

1. The initial values for this register can be loaded from the EEPROM by the management firmware after power-up reset.

4.4.3.10.10 Flexible TCO Filter Table Registers – FTFT (0x09400-0x097FC; RW)

Each of the Four Flexible TCO Filters table registers (FTFT) 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 FTFT register.

Each 128-byte filter is composed of 32 Dword entries, where each two Dwords are accompanied by an 8-bit mask, one bit per filter byte. The bytes in each two Dwords are written in host order. For example, byte0 written to bits [7:0], byte1 to bits [15:8] etc. The mask field is set so that bit0 in the mask masks byte0, bit 1 masks byte 1 etc. A value of one in the mask field means that the appropriate byte in the filter should be compared to the appropriate byte in the incoming packet.

Note: The mask field must be 8bytes aligned even if the length field is not 8 bytes aligned as the hardware implementation compares 8 bytes at a time so it should get extra masks until the end of the next Qword. Any mask bit that is located after the length should be set to zero indicating no comparison should be done.

Note: If the actual length, which is defined by the length field register and the mask bits, is not 8 bytes aligned there might be a case that a packet which is shorter than the actual required length pass the flexible filter. This can happen due to comparison of up to 7 bytes that come after the packet but are not a real part of the packet.

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. If actual packet length is less than the length specified by this field, the filter fails. Otherwise, it depends on the result of actual byte comparison. The value should not be greater than 128.

The initial values for the FTFT registers can be loaded from the EEPROM after power-up reset. The FTFT registers are written by the BMC and not accessible to the host for writing. The registers are used to filter manageability packets.

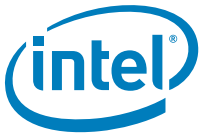
Reset – The FTFT registers are cleared on Internal Power On Reset or LAN_PWR_GOOD only.

31 8	31 8	7 0	31 0	31 0
Reserved	Reserved	Mask [7:0]	Dword 1	Dword 0
Reserved	Reserved	Mask [15:8]	Dword 3	Dword 2
Reserved	Reserved	Mask [23:16]	Dword 5	Dword 4
Reserved	Reserved	Mask [31:24]	Dword 7	Dword 6

……..

31 8	31 8	7 0	31 0	31 0
Reserved	Reserved	Mask [127:120]	Dword 29	Dword 28
Length	Reserved	Mask [127:120]	Dword 31	Dword 30

Field	Dword	Address	Bit(s)	Initial Value
Filter 0 Dword0	0	0x09400	31:0	X
Filter 0 Dword1	1	0x09404	31:0	X
Filter 0 Mask[7:0]	2	0x09408	7:0	X
Reserved	3	0x0940C		X



Filter 0 Dword2	4	0x09410	31:0	X
...				
Filter 0 Dword30	60	0x094F0	31:0	X
Filter 0 Dword31	61	0x094F4	31:0	X
Filter 0 Mask[127:120]	62	0x094F8	7:0	X
Length	63	0x094FC	6:0	X

4.4.3.11 PCIe Registers

4.4.3.11.1 PCIe Control Register – GCR (0x11000; RW)

Field	Bit (s)	Initial Value	Description
Reserved	0	0b	Reserved
Reserved	1	0b	Reserved
Reserved	2	0b	Reserved
CBMRX	3	0b	I/OAT Message Received This bit indicates that an I/OAT message was received by the 82598.
Reserved	7:4	X	Reserved
FW Self-Reset	8	0b	When set, firmware performs a self reset.
Rx_L0s_ Adjustment	9	1b	If set, the replay timer always adds the required L0s adjustment. When 0b. the replay timer adds it only when Tx L0s is active.
Reserved	11:10	00b	Reserved

Completion_ Timeout_Value	15:12	0000b[1]	Indicates the selected value for completion timeout. Decoding of this field depends on the PCIe capability version: Capability version 0x1: 0000b = 50 µs to 10 ms (default). 0001b = 10 ms to 250 ms. 0010b = 250 ms to 4 s. 0011b = 4 s to 64 s. Other = Reserved. Capability version 0x2: 0000b = 50 µs to 50 ms. 0001b = 50 µs to 100 µs. 0010b = 1 ms to 10 ms. 0011b = Reserved. 0100b = Reserved. 0101b = 16 ms to 55 ms. 0110b = 65 ms to 210 ms. 0111b = Reserved. 1000b = Reserved. 1001b = 260 ms to 900 ms. 1010b = 1 s to 3.5 s. 1011b = Reserved. 1100b = Reserved. 1101b = 4 s to 13 s. 1110b = 17 s to 64 s. 1111b = Reserved. **Note:** For Capability Version 2, this field is read only.
Completion_ Timeout_ Resend	16	1b[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.
Completion_ Timeout_ Disable	17	0b[1]	Indicates if the PCIe completion timeout is supported. 0b = Completion timeout enabled. 1b = Completion timeout disabled.
Field	**Bit (s)**	**Initial Value**	**Description**
PCIe Capability Version	18	1b[1]	Reports the PCIe capability version supported. 0b = Capability version: 0x1. 1b = Capability version: 0x2.
Reserved	19	0b	Reserved
APBACD	20	0b	Auto PBA Clear Disable When set to 0b, PBA entry is cleared on the falling edge of the appropriate interrupt request to the PCIe block.
hdr_log inversion	21	0b	If set the header log in error reporting is written as 31:0 to log1, 63:643 in log2, etc. If not, the header is written as 127:96 in log1, 95:64 in log 2, etc.
Reserved	22	1b	Reserved Must be set to 1b.
Reserved	23	0b	Reserved
L0S_Entry_ Latency	24	0b	L0s Entry Latency Set to 0b to indicate that the L0s entry latency is the same as L0s exit latency. Set to 1b to indicate that the L0s entry latency is the same as L0s Exit Latency/4.

Reserved	26:25	11b[1]	Reserved
Reserved	27	0b	Reserved
Gio_dis_rd_err	28	0b	Disable running disparity error of the PCIe 108b decoders.
Gio_good_l0s	29	0b	Force good PCIe L0s training.
Self_test_result	30	0b	If set, the self test result finished successfully.
Reserved	31	0b	Reserved

1. Initial value is loaded from the EEPROM.

4.4.3.11.2 PCIe Timer Value – GTV (0x11004; RW)

Field	Bit(s)	Initial Value	Description
RTVALUE	14:0	0x1000	Replay Timer Value Value is in units of 4 ns.
Reserved	30:15	0x0	Reserved
RTVALID	31	0b	Replay Timer Valid When set to 1b, RTVALUE is used for the timeout value for TLP packet retransmission.

4.4.3.11.3 Function-Tag Register FUNCTAG (0x11008; RW)

Field	Bit(s)	Initial Value	Description
cnt_3_tag	31:29	0x0	Tag number for event 6/1D, if located in counter 3.
cnt_3_func	28:24	0x0	Function number for event 6/1D, if located in counter 3.
cnt_2_tag	23:21	0x0	Tag number for event 6/1D, if located in counter 2.
cnt_2_func	20:16	0x0	Function number for event 6/1D, if located in counter 2.
cnt_1_tag	15:13	0x0	Tag number for event 6/1D, if located in counter 1.
cnt_1_func	12:8	0x0	Function number for event 6/1D, if located in counter 1.
cnt_0_tag	7:5	0x0	Tag number for event 6/1D, if located in counter 0.
cnt_0_func	4:0	0x0	Function number for event 6/1D, if located in counter 0.

4.4.3.11.4 PCIe Latency Timer – GLT (0x1100C; RW)

Field	Bit(s)	Initial Value	Description
LTVALUE	14:0	0x40	Latency Timer Value Value is in units of 4 ns.
Reserved	30:15	0x0	Reserved
LTVALID	31	0b	Latency Timer Valid When set to 1b, LTVALUE is used for the maximum latency before sending ACK/ NACK.

4.4.3.11.5 Function Active and Power State to Manageability – FACTPS (0x10150; RO)

This register is for use by the firmware for configuration.

Field	Bit(s)	Initial Value	Description
PM State changed	31	0b	Indication that one or more of the functions power states had changed. This bit is also a signal to the manageability unit to create an interrupt. This bit is cleared on read, and is not set for at least eight cycles after it was cleared.
LAN Function Sel	30	0b[1]	When LAN Function Sel equals 0b, LAN 0 is routed to PCI function 0 and LAN 1 is routed to PCI function 1. If the *LAN Function Sel* equals 1b, LAN 0 is routed to PCI function 1 and LAN 1 is routed to PCI function 0.
MNGCG	29	0b	Manageability Clock Gated When set indicates that the manageability clock is gated.
Reserved	28:10	00b	Reserved
Func1 Aux_En	9	0b	Function 1 *Auxiliary (AUX) Power PM Enable* bit shadow from the configuration space
LAN1 Valid	8	0b	LAN 1 Enable When this bit is 0b, it indicates that the LAN 0 function is disabled. When the function is enabled, the bit is 1b. This bit reflects if the function is disabled through the external pad
Func1 Power State	7:6	00b	Power state indication of function 1. 00b -> DR. 01b -> D0u. 10b -> D0a. 11b -> D3.
Reserved	5:4	00b	Reserved
Func0 Aux_En	3	0b	Function 0 *Auxiliary (AUX) Power PM Enable* bit shadow from the configuration space.

LAN0 Valid	2	0b	LAN 0 Enable When this bit is 0b, it indicates that the LAN 0 function is disabled. When the function is enabled, the bit is 1b. This bit reflects if the function is disabled through the external pad.
Func0 Power State	1:0	00b	Power state indication of function 0 00b -> DR. 01b -> D0u. 10b -> D0a. 11b -> D3.

1. This bit is initiated from the EEPROM.

4.4.3.11.6 PCIe Analog Configuration Register – PCIEANACTL (0x11040; RW)

This register is for use by the device hardware for configuring analog circuits in the PCIe block.

Field	Bit(s)	Initial Value	Description
Done Indication	31	1	When a write operation completes, this bit is set to 1b indicating that new data can be written. This bit is over written to 0b by new data.
Reserved	30:20	0	Reserved
Target	19:16	0	Analog target to the configuration. 0000b = Lane 0 0001b = Lane 1 0010b = Lane 2 0011b = Lane 3 0100b = Lane 4 0101b = Lane 5 0110b = Lane 6 0111b = Lane 7 1000b = All lanes 1001b = SCC PLL 1010b:1111b – Reserved
Address	15:8	0	Address to PCIe Analog registers.
Data	7:0	0	Data to PCIe Analog registers.

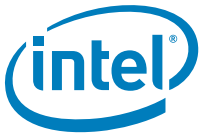
4.4.3.11.7 Software Semaphore Register – SWSM (0x10140; RW)

Field	Bit(s)	Initial Value	Description
SMBI	0	0x0	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 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 is cleared on GIO soft reset.

SWESMBI	1	0x0	Software EEPROM Semaphore bit This bit should be set only by the software device driver (read-only to firmware). The bit is not set if bit 0 in the FWSM register is set. The software device driver should set this bit and then read it to see if it was set. If it was set, it means that the software device driver can read/write from/to the EEPROM. The software device driver should clear this bit when finishing its EEPROM's access. Hardware clears this bit on GIO soft reset.
WMNG	2	0x0	Wake Manageability Clock When this bit is set the hardware wakes the manageability clock if gated. Asserting this bit does not clear the *CFG_DONE* bit in the EEMNGCTL register. This bit is self cleared on writes.
Reserved	31:3	0x0	Reserved.

4.4.3.11.8 Firmware Semaphore Register – FWSM (0x10148; RW)

Field	Bit(s)	Initial Value	Description
EEP_FW_ semaphore	0	0b	EEPROM Firmware Semaphore Firmware should set this bit to 1b before accessing the EEPROM. If software using the SWSM does not lock the EEPROM, firmware is able to set it to 1b. Firmware should set it to 0b after completing an EEPROM access.
FW_mode	3:1	0x0	Firmware Mode Indicates the firmware mode as follows: 0x0 = None (manageability Off). 0x1 = Reserved 0x2 = Pass Through (PT) mode 0x3 = Reserved 0x4 = Host interface enable only. Else = Reserved
Reserved	5:4	00b	Reserved
EEP_reload_ ind	6	0b	EEPROM Reloaded Indication Set to 1b after firmware reloaded EEPROM. Cleared by firmware once the Clear Bit host command is received from host software.
Reserved	14:7	0x0	Reserved
FW_Val_bit	15	0b	Firmware Valid Bit Hardware clears this bit in reset de-assertion so software can know firmware mode (bits 1-5) is invalid. Firmware should set it to 1b when it is ready (end of boot sequence).
Reset_cnt	18:16	0x0	Reset counter firmware increments this field after every reset.



Ext_err_ind	24:19	0x0	External Error Indication Firmware writes here the reason that the firmware has reset/clock gated (EEPROM, Flash, patch corruption, etc.). Possible values: 0x00 = No Error. 0x01 = Invalid EEPROM checksum. 0x02 = Unlocked secured EEPROM. 0x03 = Clock off host command. 0x04 = Invalid Flash checksum. 0x05 = C0 checksum failed. 0x06 = C1 checksum failed. 0x07 = C2 checksum failed. 0x08 = C3 checksum failed. 0x09 = TLB table exceeded. 0x0A = DMA load failed. 0x0B = Bad hardware version in patch load. 0x0C = Flash device not supported in the 82598. 0x0D = Unspecified error. 0x3F = Reserved – maximum error value.
PCIe_config_ err_ind	25	0b	PCIe Configuration Error Indication Set to 1b by firmware when it fails to configure PCIe interface. Cleared by firmware upon successful configuration of PCIe interface.
PHY_SerDes0_ config_ err_ind	26	0b	PHY/SerDes0 Configuration Error Indication Set to 1b by firmware when it fails to configure PHY/SerDes of LAN0. Cleared by firmware upon successful configuration of PHY/SerDes of LAN0.
Field	**Bit(s)**	**Initial Value**	**Description**
PHY_SerDes1_ config_ err_ind	27	0b	PHY/SerDes1 Configuration Error Indication Set to 1b by firmware when it fails to configure PHY/SerDes of LAN1. Cleared by firmware upon successful configuration of PHY/SerDes of LAN1.
Unlock_EEP	28	0b	Unlock EEPROM Set to 1b by software in order to enable re-writing to the EEPROM at address 0x00 (EEPROM Control Word 1). Cleared by firmware once EEPROM Control Word 1 is unlocked.
Reserved	31:29	0x0	Reserved

Note: This register should be written only by manageability firmware. The software device driver should only read this register.

Note: Firmware ignores the EEPROM semaphore in operating system hung states. Bits 15:0 are cleared on firmware reset.

4.4.3.11.9 General Software Semaphore Register – GSSR (0x10160; RW)

Field	Bit(s)	Initial Value	Description
SMBITS	9:0	0	Semaphore Bits Each bit represents a different software semaphore. Hardware implementation is read/write registers. Bits 4:0 are owned by software while bits 9:5 are owned by firmware. Hardware does not lock access to these bits.

Reserved	30:10	0	Reserved
REGSMP	31	0	Register Semaphore This bit is used to semaphore the access to this register (not hardware block). When the bit value is 0b and the register is read, the read transaction shows 0b and the bit is set (next read reads as 1b). Writing 0b to this bit clears it. A software device driver that reads this register and gets the value of 0b for this bit locks the access to this register until it clears this bit. **Note:** No hardware lock for register access.

SMBITS are reset on Internal Power On Reset or LAN_PWR_GOOD.

Software and firmware synchronize accesses to shared resources in the 82598 through a semaphore mechanism and a shared configuration register. The SWESMBI bit in the Software Semaphore (SWSM) register and the EEP_FW_semaphore bit in the Firmware Semaphore (FWSM) register serve as a semaphore mechanism between software and firmware.

Once software or firmware takes control over the semaphore, it might access the General Software Semaphore (GSSR) register and claim ownership of a specific resource. The GSSR includes pairs of bits (one owned by software and the other by firmware), where each pair of bits control a different resource. A resource is owned by software or firmware when the respective bit is set. Note that it is illegal to have both bits in a pair set at the same time.

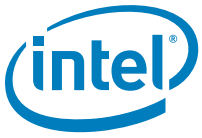
The software/firmware interface uses the following bit assignment convention for the GSSR semaphore bits.

Field	Bit	Description
SW_EEP_SM	0	When set to 1b EEPROM access is owned by software
SW_PHY_SM0	1	When set to 1b, PHY 0 access is owned by software
SW_PHY_SM1	2	When set to 1b, PHY 1 access is owned by software
SW_MAC_CSR_SM	3	When set to 1b, software owns access to shared CSRs
SW_FLASH_SM	4	Software Flash semaphore
FW_EEP_SM	5	When set to 1b, EEPROM access is owned by firmware
FW_PHY_SM0	6	When set to 1b, PHY 0 access is owned by firmware
FW_PHY_SM1	7	When set to 1b, PHY 1 access is owned by firmware
FW_MAC_CSR_SM	8	When set to 1b, firmware owns access to shared CSRs
Reserved	9	Reserved for future firmware use

When software or firmware gains control over the GSSR, it checks if a certain resource is owned by the other (the bit is set). If not, it might set its bits for that resource, taking ownership of the resource. The same process (claiming the semaphore and accessing the GSSR) is done when a resource is being freed.

The following example shows how software might use this mechanism to own a resource (firmware accesses are done in an analogous manner):

1. Software takes control over the software/firmware semaphore.



a. Software writes a 1b to the SWESMBI bit in the SWSM.

b. Software reads the SWESMBI bit. If set, software owns the semaphore. If cleared, this is an indication that firmware currently owns the semaphore. Software should retry the previous step after some delay.

2. Software reads the GSSR and checks the firmware bit in the pair of bits that control the resource is wishes to own.

   a. If the bit is cleared (firmware does not own the resource), software sets the software bit in the pair of bits that control the resource is wishes to own.

   b. If the bit is set (firmware owns the resource), go to step 4.

3. Software releases the software/firmware semaphore by clearing the SWESMBI bit in the SWSM.

4. If software did not succeed in owning the resource (from step 2b), software repeats the process after some delay.

The following example shows how software might use this mechanism to release a resource (firmware accesses are done in an analogous manner):

1. Software takes control over the software/firmware semaphore.

   a. Software writes a 1b to the SWESMBI bit in the SWSM.

   b. Software then reads the SWESMBI bit. If set, software owns the semaphore. If cleared, this is an indication that firmware currently owns the semaphore. Software should retry the previous step after some delay.

2. Software writes a 0b to the software bit in the pair of bits that control the resource is wishes to release in the GSSR.

3. Software releases the software/firmware semaphore by clearing the *SWESMBI* bit in the SWSM.

4. Software waits some delay before trying to gain the semaphore again.

The following are time periods used by firmware.

Description	Time
Time to backoff from a failed attempt to get the software/firmware semaphore to the next attempt.	5 ms
Time after which to access the GSSR register, by force, if the software/firmware semaphore is still unavailable.	10 ms
Time after which to access the EEPROM, by force, if GSSR.EEP_SM still not available.	1 s
Time after which to access PHY 0, by force, if GSSR.PHY_SM0 still not available.	1 s
Time after which to access PHY 1, by force, if GSSR.PHY_SM1 still not available.	1 s
Time after which to access the MAC CSR mechanism, by force, if GSSR.MAC_CSR_SM is still not available.	10 ms

In a similar way, the SW_FLASH_SM is used to synchronize between the two software device drivers on the Flash resource to make sure both drivers are not accessing the Flash at the same time. A software device driver that wants to access the Flash, first checks the state of the SW_FLASH_SM bit, and if set, does not access the Flash (used by the other software device). If it is cleared, the software device driver sets the semaphore and then accesses the Flash. Once the software device driver completes all Flash accesses, it releases the semaphore and enables the other software device driver to access the Flash.

4.4.3.11.10 Mirrored Revision ID- MREVID (0x11064; RO)

Field	Bit(s)	Initial Value	Description
EEPROM_RevID	7:0	0x0	Mirroring of Rev ID loaded from EEPROM.
DEFAULT_RevID	15:8	0x0	Mirroring of Default Rev ID, before EEPROM load (0x0 for the 82598 A0).
Reserved	31:16	0x0	Reserved

4.4.3.12 DCA Control Registers

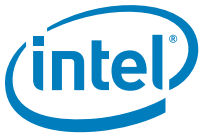
4.4.3.12.1 DCA Requester ID Information Register- DCA_ID (0x11070; R)

To ease software implementation, a DCA Requester ID field, composed of Device ID, Bus # and Function # is set up in MMIO space for software to program the chipset DCA Requester ID Authentication register.

Field	Bit(s)	Initial Value	Description
Function Number	2:0	0x0	Function Number Function number assigned to the function based on BIOS/OS enumeration.
Device Number	7:3	0x0	Device Number Device number assigned to the function based on BIOS/OS enumeration.
Bus Number	15:8	0x0	Bus Number Bus number assigned to the function based on BIOS/OS enumeration.
Reserved	31:16	0x0	Reserved

4.4.3.12.2 DCA Control Register- DCA_CTRL (0x11074; RW)

Field	Bit(s)	Initial Value	Description
DCA_DIS	0	1b	DCA Disable When 0b, DCA tagging is enabled for the 82598. When 1b, DCA tagging is disabled for the 82598.
DCA_MODE	4:1	0x0	DCA Mode When 0000b, platform is FSB. In this case, the TAG field in the TLP header is bit 0 (DCA enable) and bits 3:1 are CU ID. When 0001b, platform is CSI. In this case, when DCA is disabled for a given message, the TAG field is 11111b; if DCA is enabled, the TAG is set per queue as programmed in the relevant DCA control register. Other values are undefined.
Reserved	31:5	0x0	Reserved



4.4.3.13 MAC Registers

4.4.3.13.1 PCS_1G Global Config Register 1 – PCS1GCFIG (0x04200, RW)

Field	Bit(s)	Initial Value	Description
Reserved	31	0b	Reserved
Pcs_isolate	30	0b	PCS Isolate Setting this bit isolates the 1 Gb/s PCS logic from the MAC's data path. PCS control codes are still sent and received.
Reserved	29:0	0x0	Reserved

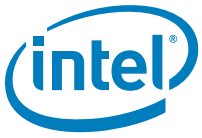
4.4.3.13.2 PCG_1G Link Control Register – PCS1GLCTL (0x04208; RW)

Field	Bit(s)	Initial Value	Description
Reserved	31:26	0x0	Reserved
Reserved	25	1b	Reserved
Reserved	24:21	0x0	Reserved
Reserved	20	0b	Reserved – must be set to 0b.
Reserved	19	0b	Reserved
AN 1G TIMEOUT EN	18	1b	Auto Negotiation1 Gb/s Timeout Enable This bit enables the 1 Gb/s auto negotiation timeout feature. During 1 Gb/s auto negotiation if the link partner doesn't respond with auto negotiation pages but continues to send good IDLE symbols then LINK UP is assumed. (This enables a link-up condition when a link partner is not auto-negotiation capable and does not affect otherwise).
AN 1G RESTART	17	0b	Auto Negotiation 1 Gb/s Restart Setting this bit restarts the clause 37 1 Gb/s auto negotiation process. This bit is self clearing.
Reserved	16	0b	Reserved
Reserved	15:7	0x0	Reserved
LINK LATCH LOW	6	0b	Link Latch Low Enable If this bit is set then Link OK going LOW (negedge) is latched till CPU read happens. Once CPU read happens, Link OK is continuously updated until Link OK again goes LOW (negedge is seen).
FORCE 1G LINK	5	0b	Force 1 Gb/s Link If this bit is set then internal LINK_OK variable is forced to *Forced Link Value*, bit 0 of this register. Else LINK_OK is decided by internal AN/SYNC state machines. This bit is only valid when the link mode is 1 Gb/s.

Reserved	4:1	0x0	Reserved
FLV	0	0b	Forced Link 1 Gb/s Value This bit denotes the link condition when force link is set. 0b = Forced Link down. 1b = Forced 1 Gb/s link up.

4.4.3.13.3 PCS_1G Link Status Register – PCS1GLSTA (0x0420C; RO)

Field	Bit(s)	Initial Value	Description
Reserved	31:21	0x0	Reserved.
AN ERROR (RW)	20	0b	Auto Negotiation Error This bit indicates that an auto negotiation error condition was detected in 1 Gb/s auto negotiation mode. Valid after the *AN 1G Complete* bit is set. Auto negotiation error conditions: • Both node not full duplex or remote fault indicated or received. • Software can also force an auto negotiation error condition by writing to this bit (or can clear a existing auto negotiation error condition). Cleared at the start of auto negotiation.
Reserved	19	0b	Reserved
AN 1G TIMEDOUT	18	0b	Auto Negotiation1 Gb/s Timed Out This bit indicates 1 Gb/s auto negotiation process was timed out. Valid after *AN 1G Complete* bit is set.
Reserved	17	0b	Reserved
AN 1G COMPLETE	16	0b	Auto Negotiation1 Gb/s Complete This bit indicates that the 1 Gb/s auto negotiation process has completed.
Reserved	15:5	0x0	Reserved.
SYNC OK 1G	4	0b	Sync OK 1 Gb/s This bit indicates the current value of SYN OK from the 1 Gb/s PCS Sync state machine.
Reserved	3:1	111b	Reserved
Link_OK_1G	0	0b	Link OK 1 Gb/s This bit denotes the current 1 Gb/s Link OK status. 0b = 1 Gb/s link down. 1b = 1 Gb/s link up/ok.



4.4.3.13.4 PCS_1 Gb/s Auto Negotiation Advanced Register PCS1GANA (0x04218; RW)

Field	Bit(s)	Initial Value	Description
Reserved	31:16	0x0	Reserved
NEXTP	15	0b	Next Page Capable The 82598 asserts this bit to request next page transmission. Clear this bit when the 82598 has no subsequent next pages.
Reserved	14	0b	Reserved
RFLT	13:12	00b	Remote Fault The 82598's remote fault condition is encoded in this field. The 82598 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	11:9	0x0	Reserved
ASM	8:7	11b	ASM_DIR/PAUSE - Local Pause Capabilities The 82598's pause capability is encoded in this field. 00b = No pause. 01b = Symmetric pause. 10b = Asymmetric pause toward link partner. 11b = Both symmetric and asymmetric pause toward the 82598.
Reserved	6	0b	Reserved
FDC	5	1b	FD – Full-Duplex Setting this bit indicates that the 82598 is capable of full-duplex operation. This bit should be set to 1b for normal operation.
Reserved	4:0	0x0	Reserved

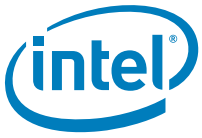
4.4.3.13.5 PCS_1GAN LP Ability Register – PCS1GANLP (0x0421C; RO)

Field	Bit(s)	Initial Value	Description
Reserved	31:16	0x0	Reserved
LPNEXTP	15	0b	LP Next Page Capable (SerDes) The link partner asserts this bit to indicate its ability to accept next pages.
ACK	14	0b	Acknowledge (SerDes) The link partner has acknowledge page reception.

PRF	13:12	00b	LP Remote Fault (SerDes)[13:12] 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.
Reserved	11:9	0x0	Reserved
LPASM	8:7	00b	LPASMDR/LPPAUSE(SerDes) The link partner's pause capability is encoded in this field. 00b = No pause. 01b = Symmetric pause. 10b = Asymmetric pause toward link partner. 11b = Both symmetric and asymmetric pause toward the 82598.
LPHD	6	0b	LP Half-Duplex (SerDes) When 1b, link partner is capable of half duplex operation. When 0b, link partner is incapable of half duplex mode.
LPFD	5	0b	LP Full-Duplex (SerDes) When 1b, link partner is capable of full duplex operation. When 0b, link partner is incapable of full duplex mode.
Reserved	4:0	0x0	Reserved

4.4.3.13.6 PCS_1G Auto Negotiation Next Page Transmit Register – PCS1GANNP (0x04220; RW)

Field	Bit(s)	Initial Value	Description
Reserved	31:16	0x0	Reserved
NXTPG	15	0b	Next Page This bit is used to indicate whether or not this is the last next page to be transmitted. The encodings are: 0b = Last page. 1b = Additional next pages follow.
Reserved	14	0b	Reserved
PGTYPE	13	0b	Message/ Unformatted Page This bit is used to differentiate a message page from an unformatted page. The encodings are: 0b = Unformatted page. 1b = Message page.
ACK2	12	0b	Acknowledge2 Acknowledge is used to indicate that the 82598 has successfully received its link partners' Link Code Word.



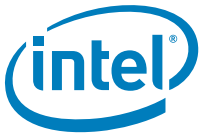
TOGGLE	11	0b	Toggle This bit is used to ensure synchronization with the link partner during next page exchange. This bit always takes the opposite value of the *Toggle* bit in the previously exchanged Link Code Word. The initial value of the *Toggle* bit in the first next page transmitted is the inverse of bit 11 in the base Link Code Word and therefore might assume a value of 0b or 1b. The *Toggle* bit is set as follows: 0b = Previous value of the transmitted Link Code Word equaled 1b. 1b = Previous value of the transmitted Link Code Word equaled 0b.
CODE	10:0	0x0	Message/Unformatted Code Field The *Message Field* is a 11-bit wide field that encodes 2048 possible messages. *Unformatted Code Field* is a 11-bit wide field that might contain an arbitrary value.

4.4.3.13.7 PCS_1G Auto Negotiation LP's Next Page Register – PCS1GANLPNP (0x04224; RO)

Field	Bit(s)	Initial Value	Description
Reserved	31:16	0x0	Reserved
NXTPG	15	0b	Next Page This bit is used to indicate whether or not this is the last next page to be transmitted. The encodings are: 0b = Last page. 1b = Additional next pages follow.
ACK	14	0b	Acknowledge The link partner has acknowledge next page reception.
MSGPG	13	0b	Message Page This bit is used to differentiate a message page from an unformatted page. The encodings are: 0b = Unformatted page. 1b = Message page.
ACK2	12	0b	Acknowledge Acknowledge is used to indicate that the 82598 has successfully received its link partners' Link Code Word.
TOGGLE	11	0b	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 might assume a value of 0b or 1b. The *Toggle* bit is set as follows: 0b = Previous value of the transmitted Link Code Word equaled 1b. 1b = Previous value of the transmitted Link Code Word equaled 0b.
CODE	10:0	0x0	Message/Unformatted Code Field The *Message Field* is a 11-bit wide field that encodes 2048 possible messages. *Unformatted Code Field* is a 11-bit wide field that might contain an arbitrary value.

4.4.3.13.8 Flow Control 0 Register – HLREG0 (0x04240, RW)

Field	Bit(s)	Initial Value	Description
TXCRCEN	0	1b	Tx CRC Enable Enables a CRC to be appended to a TX packet if requested. 1b = Enable CRC. 0b = No CRC appended, packets always passed unchanged. This bit must be set to 1b if the 82598 is enabled to send flow control frames.
RXCRCSTRP	1	1b	Rx CRC Strip Causes the CRC to be stripped from all packets 1b = Strip CRC. 0b = No CRC.
JUMBOEN	2	0b	Jumbo Frame Enable Allows frames up to the size specified in the MHADD (31:16) register. 1b = Enable jumbo frames. 0b = Disable jumbo frames.
Reserved	9:3	1111111b	Reserved and must be set to 1111111b.
TXPADEN	10	1b	Tx Pad Frame Enable Pad short Tx frames to 64 bytes if requested. 1b = Pad frames. 0b = Transmit short frames with no padding.
Reserved	11	1b	Reserved Must be set to 1b.
Reserved	12	0b	Reserved This bit should not be set to 1b.
Reserved	13	1b	Reserved
Reserved	14	0b	Reserved This bit should not be set to 1b.
LPBK	15	0b	Loopback Turn on loopback where transmit data is sent back through the receiver. To activate loopback the link should be active or AUTOC.FLU should be set. 1b = Loopback enabled. 0b = Loopback disabled.
MDCSPD	16	0b	MDC Speed High or low speed MDC to PCS, XGXS, WIS, etc. When at 10 Gb/s: 1b = 24 MHz. 0b = 2.4 MHz. When at 1Gb/s: 1b = 2.4 MHz. 0b = 240 KHz.



Field	Bit(s)	Initial Value	Description
CONTMDC	17	0b	Continuous MDC Turn off MDC between MDIO packets 1b = Continuous MDC. 0b = MDC off between packets.
Reserved	18	0b	Reserved This bit should not be set to 1b.
Reserved	19	0b	Reserved
PREPEND	23:20	0x0	Prepend Value Number Of 32-bit words, starting after the preamble and SFD, to exclude from the CRC generator and checker.
Reserved	24	0x0	Reserved
Reserved	26:25	00b	Reserved
RXLNGTHERREN	27	1b	Rx Length Error Reporting: 1b = Enable reporting of rx_length_err events if length field <0x0600. 0b = Disable reporting of all rx_length_err events.
Reserved	28	0b	Reserved
Reserved	31:29	0x0	Reserved

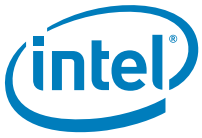
4.4.3.13.9 Flow Control Status 1 Register- HLREG1 (0x04244, RO)

Field	Bit(s)	Initial Value	Description
REVID	3:0	0001b	REV ID Version number of the MAC core. Current version (release 1.6) = 0001b.
AUTOSCAN	4	0b	Autoscan in Progress Autoscan enabled and cleared after completion. 1b = Autoscan in progress. 0b = Autoscan idle (default).
RXERRSYM	5	0b	Rx Error Symbol Error symbol during Rx packet (latch high; clear on read). 1b = Error symbol received. 0b = No error symbol.
RXILLSYM	6	0b	Rx Illegal Symbol Illegal symbol during Rx packet (latch high; clear on read). 1b = Illegal symbol received. 0b = No illegal symbol received.

Field	Bit(s)	Initial Value	Description
RXIDLERR	7	0b	Rx Idle Error Non idle symbol during idle period (latch high; clear on read). 1b = Idle error received. 0b = No idle errors received.
RXLCLFLT	8	0b	Rx Local Fault Fault reported from PMD, PMA, or PCS (latch high; clear on read). 1b = Local fault is or was active. 0b = No Local fault.
RXRMTFLT	9	0b	Rx Remote Fault Link partner reported remote fault (latch high; clear on read). 1b = Remote fault is or was active. 0b = No remote fault.
Reserved	31:10	0x0	Reserved

4.4.3.13.10 Pause and Pace Register – PAP (0x04248, RW)

Field	Bit(s)	Initial Value	Description
ReservedTXPAUSECNT	15:0	0xFFFF	Reserved TX PAUSE COUNT : Pause Count Value In Pause Quanta (a time slot of 512 bit times) Included In The TX Pause Frame Used To Pause Link Partner's Transmitter (Default = FFFF Hex)
PACE	19:16	0000b	0000b = 10 Gb/s (LAN). 0001b = 1 Gb/s. 0010b = 2 Gb/s. 0011b = 3 Gb/s. 0100b = 4 Gb/s. 0101b = 5 Gb/s. 0110b = 6 Gb/s. 0111b = 7 Gb/s. 1000b = 8 Gb/s. 1001b = 9 Gb/s. 1111b = 9.294196 Gb/s (WAN).
Reserved	31:20	0x0	Reserved



4.4.3.13.11 MDI Auto-Scan Command and Address – MACA (0x0424C; RW)

Field	Bit(s)	Initial Value	Description
ATSCNADD	15:0	0000b	Auto-Scan Address Address used for indirect address management frame format.
REGBITSEL	19:16	0000b	Register Bit Select Select one of 16 data bits to latch from the register.
DEVADD	24:20	00000b	Device Type/Register Address Five bits representing either the device type with ST = 00b or the register address with ST = 01b.
STCODE	26:25	01b	ST Code Two bits identifying start of frame and old or new protocol. 00b = New protocol. 01b = Old protocol. 1Xb = Illegal.
ATSCANEN	27	0b	Auto-Scan Enable Enable repetitive auto-scan function. 1b = Autoscan enable. 0b = Autoscan disable.
ATSCANINTEN	28	0b	Auto-Scan Interrupt Enable Enable auto-scan interrupt generation. 1b = Autoscan interrupt enable. 0b = Autoscan interrupt disable.
ATSCANINTLVL	29	1b	Auto-Scan Interrupt Level Active level for the auto-scan interrupt. 1b = Auto-scan interrupt active high. 0b = Auto-scan interrupt active low.
Reserved	31:30	00b	Reserved

4.4.3.13.12 Auto-Scan PHY Address Enable – APAE (0x04250; RW)

Field	Bit(s)	Initial Value	Description
ATSCANPHYADDEN	31:0	00b	Auto-Scan PHY Address Enable Bit to enable corresponding PHY address. 01b = Enable this address. 00b = Disable this address

4.4.3.13.13 Auto-Scan Read Data – ARD (0x04254; RW)

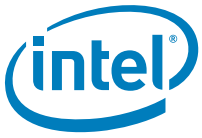
Field	Bit(s)	Initial Value	Description
RDDATA	31:0	0x0	Read Data Data from the selected register bit in the corresponding PHY.

4.4.3.13.14 Auto-Scan Interrupt Status – AIS (0x04258; RW)

Field	Bit(s)	Initial Value	Description
INTSTA	31:0	0x0	Interrupt Status Selected bit changed in corresponding PHY (latch high, clear on read). 0x1 = Data changed in corresponding PHY. 0x0 = No change in data from corresponding PHY. Register is latched high and cleared on read.

4.4.3.13.15 MDI Single Command and Address – MSCA (0x0425C; RW)

Field	Bit(s)	Initial Value	Description
MDIADD	15:0	0x0000	MDI Address Address used for new protocol MDI accesses (default = 0x0000).
DEVADD	20:16	0x0	Device Type/Reg Address Five bits representing to either the device type with ST = 00b or the register address with ST = 01b.
PHYADD	25:21	0x0	PHY Address The address of the external device.
OPCODE	27:26	00b	OP Code Two bits identifying operation to be performed (default = 00b). 00b = Address cycle (new protocol only). 01b = Write operation. 10b = Read operation. 11b = Read, increment address (new protocol only).
STCODE	29:28	01b	ST Code Two bits identifying start of frame and old or new protocol (default = 01b). 00b = New protocol. 01b = Old protocol. 1x = Illegal.
MDICMD	30	0b	MDI Command Perform the MDI operation in this register; cleared when done. 1b = Perform operation; operation in progress. 0b = MDI ready; operation complete (default).
MDIINPROGEN	31	0b	MDI In Progress Enable Generate MDI in progress when operation completes. 1b = MDI in progress enabled. 0b = MDI ready disable (default).



4.4.3.13.16 MDI Single Read and Write Data – MSRWD (0x04260; RW)

Field	Bit(s)	Initial Value	Description
MDIWRDATA	15:0	0x0000	MDI Write Data Write data For MDI writes to the external device.
MDIRDDATA	31:16	0x0000	MDI Read Data Read data from the external device (RO).

4.4.3.13.17 Low MAC Address – MLADD (0x04264; RW)

Field	Bit(s)	Initial Value	Description
MACL	31:0	0x0	MAC ADRESS [31:0] Least significant 32 bits of the MAC address

4.4.3.13.18 MAC Address High and Max Frame Size – MHADD (0x04268; RW)

Field	Bit(s)	Initial Value	Description
ReservedMACH	15:0	0x0	Reserved MAC ADRESS [47:32] Most significant 16 bits of the MAC address
MFS	31:16	0x5EE	MAX Frame Size Maximum number of bytes in a frame that can be transmitted. Sets the boundary between oversize and jumbo frames on receive when jumbo frames are enabled. The value includes the CRC. **Note:** For the receive side, if the packet has a VLAN field then the value of the MFS is internally increased by four bytes. **Note:** For the transmit side, enforcing the MAX frame size restriction should be done by software (the 82598 does not limit the transmit packet size).

4.4.3.13.19 XGXS Status 1 – PCSS1 (0x4288; RO)

Field	Bit(s)	Initial Value	Description
Reserved	31:8	0x0	Reserved
Local fault	7	1b	1b = LF detected on transmit or receive path. The LF bit is set to 1b when either of the local fault bits located in PCS Status 2 register are set to 1b. 0b = No LF detected on receive path.
Reserved	6:3	0x0	Reserved

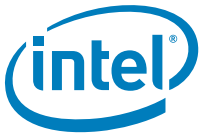
PCS receive link status	2	0b	1b = PCS receive link up. For 10BASE-X ->lanes de-skewed. 0b = PCS receive link down. The receive link status remains cleared until it is read (latching low).
Reserved	1:0	00b	Reserved

4.4.3.13.20 XGXS Status 2 – PCSS2 (0x0428C; RO)

Field	Bit(s)	Initial Value	Description
Reserved	31:16	0x0	Reserved
Device present	15:14	10b	10b = 82598 responding at this address. 11b, 01b, 00b = No 82598 responding at this address.
Reserved	13:12	00b	Reserved
Transmit local fault	11	0b	1b = Local fault condition on the transmit path. 0b = No local fault condition on the transmit path (latch high).
Receive local fault	10	1b	1b = Local fault condition on the receive path. 0b = No local fault condition on the receive path (latch high).
Reserved	9:3	0x0	Reserved
10GBASE-W capable	2	0b	1b = PCS is able to support 10GBASE-W port type. 0b = PCS is not able to support 10GBASE-W port type.
10GBASE-X capable	1	1b	1b = PCS is able to support 10GBASE-X port type. 0b = PCS is not able to support 10GBASE-X port type.
10GBASE-R capable	0	0b	1b = PCS is able to support 10GBASE-R port type. 0b = PCS is not able to support 10GBASE-R port type.

4.4.3.13.21 10GBASE-X PCS Status – XPCSS (0x04290; RO)

Field	Bit(s)	Initial Value	Description
Reserved	31:30	00b	Reserved
Lane 3 Signal Detect	29	0b	1b = Indicates a signal is detected. 0b = Indicates noise, no signal is detected.
Lane 2 Signal Detect	28	0b	1b = Indicates a signal is detected. 0b = Indicates noise, no signal is detected.
Lane 1 Signal Detect	27	0b	1b = Indicates a signal is detected. 0b = Indicates noise, no signal is detected.
Lane 0 Signal Detect	26	0b	1b = Indicates a signal is detected. 0b = Indicates noise, no signal is detected.



Lane 3 comma count 4	25	0b	1b = Indicates the comma count for that lane has reached four. 0b = Indicates the comma count for that lane is less than four (latch high).
Lane 2 comma count 4	24	0b	1b = Indicates the comma count for that lane has reached four. 0b = Indicates the comma count for that lane is less than four (latch high).
Lane 1 comma count 4	23	0b	1b = Indicates the comma count for that lane has reached four. 0b = Indicates the comma count for that lane is less than four (latch high).
Lane 0 comma count4	22	0b	1b = Indicates the comma count for that lane has reached four. 0b = Indicates the comma count for that lane is less than four (latch high).
Lane 3 invalid code	21	0b	1b = Indicates an invalid code was detected for that lane. 0b = Indicates no invalid code was detected (latch high).
Lane 2 invalid code	20	0b	1b = Indicates an invalid code was detected for that lane 0b = Indicates no invalid code was detected (latch high).
Lane 1 invalid code	19	0b	1b = Indicates an invalid code was detected for that lane. 0b = Indicates no invalid code was detected (latch high).
Lane 0 invalid code	18	0b	1b = Indicates an invalid code was detected for that lane. 0b = Indicates no invalid code was detected (latch high).
Align column count 4	17	0b	1b = Indicates the align column count has reached four. 0b = Indicates the align column count is less than four (latch high).
De-skew error	16	0b	1b = Indicates a de-skew error was detected. 0b = Indicates no de-skew error was detected (latch high).
Reserved	15:13	0x0	Reserved, ignore when read.
10GBASE-X lane alignment status	12	0b	1b = 10GBASE-X PCS receive lanes aligned (align_status = ok). 0b = 10GBASE-X PCS receive lanes not aligned.
Reserved	11: 4	0x0	Ignore when read.
Lane 3 sync	3	0b	1b = Lane 3 is synchronized. 0b = Lane 3 is not synchronized.
Lane 2 sync	2	0b	1b = Lane 2 is synchronized. 0b = Lane 2 is not synchronized.
Field	**Bit(s)**	**Initial Value**	**Description**
Lane 1 sync	1	0b	1b = Lane 1 is synchronized. 0b = Lane 1 is not synchronized.
Lane 0 sync	0	0b	1b = Lane 0 is synchronized. 0b = Lane 0 is not synchronized.

4.4.3.13.22 SerDes Interface Control Register – SERDESC (0x04298; RW)

Field	Bit(s)	Initial Value	Description
swap_rx_lane_0	31:30	00b[1]	Determines which core lane is mapped to MAC rx lane 0 00b = Core Rx lane 0 to MAC Rx lane 0. 01b = Core Rx lane 1 to MAC Rx lane 0. 10b = Core Rx lane 2 to MAC Rx lane 0. 11b = Core Rx lane 3 to MAC Rx lane 0.
swap_rx_lane_1	29:28	01b[1]	Determines which core lane is mapped to MAC Rx lane 1.
swap_rx_lane_2	27:26	10b[1]	Determines which core lane is mapped to MAC Rx lane 2.
swap_rx_lane_3	25:24	11b[1]	Determines which core lane is mapped to MAC Rx lane 3.
swap_tx_lane_0	23:22	00b[1]	Determines the core destination Tx lane for MAC Tx Lane 0 00b = MAC tx lane 0 to core tx lane 0. 01b = MAC tx lane 0 to core tx lane 1. 10b = MAC tx lane 0 to core tx lane 2. 11b = MAC tx lane 0 to core tx lane 3.
swap_tx_lane_1	21:20	01b[1]	Determines core destination Tx lane for MAC Tx lane 1.
swap_tx_lane_2	19:18	10b[1]	Determines core destination Tx lane for MAC Tx lane 2.
swap_tx_lane_3	17:16	11b[1]	Determines core destination Tx lane for MAC Tx lane 3.
Reserved	15:12	0x0[1]	Reserved Software should not change the default EEPROM value.
Reserved	11:8	0x0[1]	Reserved Software should not change the default EEPROM value.
Field	**Bit(s)**	**Initial Value**	**Description**

Rx_lanes_polarity	7:4	0x0[1]	Bit 7 – Changes bits polarity of MAC Rx lane 3 Bit 6 – Changes bits polarity of MAC Rx lane 2 Bit 5 – Changes bits polarity of MAC Rx lane 1 Bit 4 – Changes bits polarity of MAC Rx lane 0 Changes bits polarity if set to 0x1
Tx_lanes_polarity	3:0	0x0[1]	Bit 3 – Changes bits polarity of mac Tx lane 3. Bit 2 – Changes bits polarity of mac Tx lane 2. Bit 1 – Changes bits polarity of mac Tx lane 1. Bit 0 – Changes bits polarity of mac Tx lane 0. Changes bits polarity if set to 0x1.

1. Loaded from the EEPROM.

4.4.3.13.23 FIFO Status/CNTL Report Register – MACS (0x0429C; RW)

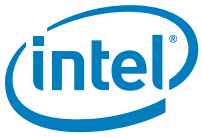
This register reports FIFO status in xgmii_mux.

Field	Bit(s)	Initial Value	Description
Rx FIFO overrun	31	0b	Indicates FIFO overrun in xgmii_mux_rx_fifo.
Rx FIFO underrun	30	0b	Indicates FIFO underrun in xgmii_mux_rx_fifo.
Tx FIFO overrun	29	0b	Indicates FIFO overrun in xgmii_mux_tx_fifo.
Tx FIFO underrun	28	0b	Indicates FIFO underrun in xgmii_mux_tx_fifo.
Config FIFO threshold	27:24	0x6	Determines threshold for asynchronous FIFO (generation of data_available signal is determined by cfg_fifo_th[3:0]).
Reserved	23:16	0x7	Reserved
Reserved	15:4	0x0	Reserved
Reserved	3	0b	Reserved
Reserved	2	0b	Reserved
Reserved	1	0b	Reserved
Reserved	0	0b	Reserved

4.4.3.13.24 Auto Negotiation Control Register – AUTOC (0x042A0; RW)

Field	Bit(s)	Initial Value	Description
KX_support	31:30	11b[1]	The value shown in bits A0:A1 of the *Technology Ability* field of the auto negotiation word. 00b: Illegal value. 01b: A0 = 1 A1 = 0. KX supported. KX4 not supported. 10b: A0 = 0 A1 = 1. KX not supported. KX4 supported. 11b: A0 = 1 A1 = 1. KX supported. KX4 supported.

PB	29:28	00b[1]	Pause Bits The value of these bits is loaded to bits D11-D10 of the link code word (pause data). Bit 29 is loaded to D11.
RF	27	0b[1]	This bit is loaded to the RF of the auto negotiation word.
ANPDT	26:25	00b[1]	Auto Negotiation Parallel Detect Timer Configure the parallel detect counters. 00b = 1 ms. 01b = 2 ms. 10b = 5 ms 11b = 8 ms.
Reserved	24	1b[1]	Reserved Must be set to 0b for normal operation.
ANRXDM	23	1b[1]	Auto Negotiation RX Drift Mode Enable following the drift caused by PPM in the RX data. 0b = Disable drift mode. 1b = Enable drift mode.
ANRXAT	22:18	0x3[1]	Auto Negotiation RX Align Threshold Set threshold to determine alignment is stable.
Reserved	17:16	00b[1]	Reserved
LMS	15:13	001b[1]	Link Mode Select Selects the active link mode: 000b = 1 Gb/s link (no auto negotiation). 001b = 10 Gb/s link (no auto negotiation). 010b = 1 Gb/s link with clause 37 auto negotiation enable. 011b = Reserved. 100b = KX4/KX auto negotiation disable. 1 Gb/s (Clause 37) auto negotiation disable. 101b = Reserved. 110b = KX4/KX auto negotiation enable. 1 Gb/s (Clause 37) auto negotiation enable. 111b = Reserved.
Restart_AN	12	0b[1]	Restart KX/KX4 Auto Negotiation process (self-clearing bit) 0b = No action needed. 1b = Restart KX/KX4 auto negotiation.
Field	**Bit(s)**	**Initial Value**	**Description**
RATD	11	0b[1]	Restart Auto Negotiation on Transition to Dx This bit enables the functionality to restart KX/KX4 auto negotiation transition to Dx(Dr/D3). 0b = Do not restart auto negotiation when the 82598 moves to the Dx state. 1b = Restart auto negotiation to reach a low-power link mode (1 Gb/s link) when the 82598 transitions to the Dx state.



D10GMP	10	0b[1]	Disable 10 Gb/s (KX4) on Dx(Dr/D3) Without Main Power. 0b = No specific action. 1b = Disable 10 Gb/s when main power is removed. When RATD bit is also set to 1b, it causes the link mode to disable 10 Gb/s capabilities and restart auto negotiation (if enabled) when the main-power (MAIN_PWR_OK) is removed.
1G_PMA_PMD	9	1b[1]	PMA/PMD Used for 1 Gb/s. 0b = BX PMA/PMD. 1b = KX PMA/PMD.
10G_PMA_PMD	8:7	10b[1]	PMA/PMD Used for 10 Gb/s. 00b = XAUI PMA/PMD. 01b = KX4 PMA/PMD. 10b = CX4 PMA/PMD. 11b = Reserved.
ANSF	6:2	00001b[1]	AN Selector This value is used as the *Selector* field in the link control word during the clause 73 auto-negotiation process. Note that the default value is according to 802.3ap draft 2.4 specification.
ANACK2	1	0b[1]	AN ACK2 This value is transmitted in the *Acknowledge2* field of the null next page that is transmitted during a next page handshake.
FLU	0	0b	Force Link Up This setting forces the auto-negotiation arbitration state machine to AN_GOOD and sets the link-up indication regardless of the XGXS/ PCS_1G status. 0b = Normal mode. 1b = MAC forced to link up. Link is active in the speed configured in AUTOC.LMS.

1. Loaded from the EEPROM.

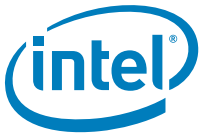
4.4.3.13.25 Link Status Register – LINKS (0x042A4; RO)

Field	Bit(s)	Initial Value	Description
KX/KX4 AN Completed	31	0b	Indicate KX/KX4 auto negotiation has completed successfully.
Link Up	30	0b	Link Up. 1b = Link is up. 0b = Link is down.
Link Speed	29	1b	Speed of the MAC link. 1b = 10 Gb/s link speed. 0b = 1 Gb/s link speed.
Reserved	28:27	01b	Reserved
Reserved	26:25	01b	Reserved
Reserved	24:23	01b	Reserved

10G link Enabled (XGXS)	22	0b	XGXS Enabled for 10 Gb/s operation.
1G link Enabled PCS_1G	21	0b	PCS_1G Enabled for 1 Gb/s operation.
1G AN enabled (clause 37 AN)	20	0b	PCS_1 Gb/s auto negotiation is enabled (clause 37).
KX/KX4 AN Receiver Idle	19	0b	KX/KX4 Auto Negotiation RX Idle 0b = Receiver is operational. 1b = Receiver is in idle- waiting to align and sync on DME.
1G Sync Status	18	1b	1G sync_status 0b = Sync_status is not synchronized to code-group. 1b = Sync_status is synchronized to code-group.
10G Align Status	17	1b	10 Gb/s align_status 0b = Align_status is not operational (deskew process not complete). 1b = Align_status is operation (all lanes are synchronized and aligned).
10G lane sync_status	16:13	1b	10 Gb/s sync_status of the lanes. Bit[16,15,14,13] show lane <3,2,1,0> status accordingly. per each bit: 0b = Sync_status is not synchronized to code-group. 1b = Sync_status is synchronized to code-group.
Transmit Local Fault	12	1b	XGXS Local Fault Sequences Transmission 0b = LF is not transmitted. 1b = LF is now transmitted.
Signal Detect	11:8	0x0	Signal Detection Bit[11, 10, 9, 8] show lane <3,2,1,0> status accordingly per each bit: 0b = A signal is not present. 1b = A signal is present.
Link Status Check	7	0b	1b = Link is up and there was no link down from last time read. 0b = Link is/was down. Latched low upon link down. Self-cleared upon read.
Field	**Bit(s)**	**Initial Value**	**Description**
KX/KX4 AN Page Received	6	0b	KX/KX4 Auto Negotiation Page Received A new link partner page was received during auto negotiation process. Latch high; clear on read.
KX/KX4 AN Next Page Received	5	0x0	KX/KX4 Auto Negotiation Next Page Received A new link partner next page was received during an auto-negotiation process.
Reserved	4:0	0x0	Reserved

4.4.3.13.26 Auto Negotiation Control 2 Register – AUTOC2 (0x042A8; RW)

Field	Bit(s)	Initial Value	Description
Reserved	31	0b	Reserved



PDD	30	0b[1]	Disable the parallel detect part in the KX/KX4 auto negotiation. When set to 1b, the auto negotiation process avoids any parallel detect activity and relies only on the DME pages receive and transmit. 1b = Reserved. 0b = Enable the parallel detect (normal operation).
Reserved	29:28	00b[1]	Reserved
Reserved	27:24	0000b[1]	Reserved
LH1GAI	23	0b[1]	Latch High 10 Gb/s Aligned Indication Override any de-skew alignment failures in the 10 Gb/s link (by latching high). This keeps the link up after first time it reached the AN_GOOD state in 10 Gb/s (unless RestartAN is set).
Reserved	21:20	00b	Reserved
Reserved	19:16	0x0	Reserved
AN Page D Low Override	15:0	0x0	Set Auto-Negotiation Advertisement Page Fields D[15:0] Used when AUTO2.AN_advertsement_page_override is set. Bits: 15 = NP. 14 = Acknowledge. 13 = RF. 12:10 = Pause C[2:0]. 9:5 = Echoed nonce field. 4:0 = Selector field.

1. Loaded from the EEPROM.

4.4.3.13.27 Auto Negotiation Control 3 Register – AUTOC3 (0x042AC; RW)

Field	Bit(s)	Initial Value	Description
Technology Ability Field High override	31:16	0x0	Set auto negotiation advertisement page fields A[26:11]. Used when AUTOC2.AN_advertisement_page_override is set.
Technology Ability Field Low override	15:5	0x0	Set auto negotiation advertisement page fields A[10:0]. Used when AUTOC2.AN_advertisement_page_override is set. **Note:** AUTOC/KX_support value must be aligned with bits [6:5] that represent A[1:0] in the override values.
Transmitted Nonce Field override	4:0	0x0	Set auto negotiation advertisement page fields T[4:0]. Used when AUTOC2.AN_advertisement_page_override is set.

4.4.3.13.28 Auto Negotiation Link Partner Link Control Word 1 Register – ANLP1 (0x042B0; RO)

Field	Bit(s)	Initial Value	Description

Reserved	31:20	0x0	Reserved
Reserved	19:16	0x0	Reserved
LP AN Page D Low	15:0	0x0	Link Partner (LP) Auto Negotiation Advertisement Page Fields D[15:0]. [15] = NP. [14] = Acknowledge. [13] = RF. [12:10] = Pause C[2:0]. [9:5] = Echoed Nonce Field. [4:0] = Selector Field.

To ensure software's ability to read the same Link Partner Link Control Word (located across two registers), once ANLP1 is read the ANLP2 register is locked until the ANLP2 register is read. ANLP2 does not hold valid data before ANLP1 is read.

4.4.3.13.29 Auto Negotiation Link Partner Link Control Word 2 Register – ANLP2 (0x042B4; RO)

Field	Bit(s)	Initial Value	Description
LP Technology Ability Field High	31:16	0x0	LP Auto Negotiation Advertisement Page Fields A[26:11].
LP Technology Ability Field Low	15:5	0x0	LP Auto Negotiation Advertisement Page Fields A[10:0].
LP Transmitted Nonce Field	4:0	0x0	LP Auto Negotiation Advertisement Page Fields T[4:0].

To ensure software's ability to read the same Link Partner Link Control Word (located across two registers), once ANLP1 is read the ANLP2 register is locked until the ANLP2 register is read. ANLP2 does not hold valid data before ANLP1 is read.

4.4.3.13.30 MAC Manageability Control Register – MMNGC (0x042D0; Host-RO/MNG-RW)

Field	Bit(s)	Initial Value	Description
Reserved	31:1	0x0	Reserved
MNG_VETO	0	0b	MNG_VETO (default 0) access read/write by manageability, read only to the host. 0b = No specific constraints on link from manageability. 1b = Hold off any low-power link mode changes. This is done to avoid link loss and interrupting manageability activity.

4.4.3.13.31 Auto Negotiation Link Partner Next Page 1 Register – ANLPNP1 (0x042D4; RO)

Field	Bit(s)	Initial Value	Description
LP AN Next Page Low	31:0	0x0	LP AN Next Page Fields D[31:0] [31:16] = Unformatted Code. [15] = NP. [14] = Acknowledge. [13] = MP. [12] = Acknowledge2. [11] = Toggle. [10:0] = Message/Unformatted Code.

To ensure software's ability to read the same Link Partner Next Page (located across 2 registers), once ANLPNP1 is read the ANLPNP2 register is locked until the ANLPNP2 register is read. ANLPNP2 does not hold valid data before ANLPNP1 is read.

4.4.3.13.32 Auto Negotiation Link Partner Next Page 2 Register – ANLPNP2 (0x042D8; RO)

Field	Bit(s)	Initial Value	Description
Reserved	31:16	0x0	Reserved
LP AN Next Page High	15:0	0x0	LP AN Next Page Fields D[47:32]. [15:0] = Unformatted Code.

To ensure software's ability to read the same Link Partner Link Control Word (located across 2 registers), once ANLPNP1 is read the ANLPNP2 register is locked until the ANLPNP2 register is read. ANLPNP2 does not hold valid data before ANLPNP1 is read.

4.4.3.13.33 Core Analog Configuration Register - ATLASCTL (0x04800; RW)

Field	Bit(s)	Initial Value	Description
Reserved	31:17	0b	Reserved
Latch Address	16	0b	0b = Normal write operation. 1b = Latch this address for next read transaction. The Data is ignored and is not written on this transaction.
Address	15:8	0b	Address to core analog registers
Data	7:0	0b	Data to core Analog registers. Data is ignored when bit 16 is set

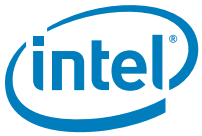
Reading the core registers must be done using the following steps:

1. Send a write command with bit 16 set, and the desired reading offset in the *Address* field (bits [15:8]).
2. Send a read command to the ATLASCTL. The returned data is from the indirect address in the core register space which was provided in step (1).



To configure (write) registers in the core block the driver should write the proper address to the ATLASCTL.Address and the data to be written to the ATLASCTL.Data.

§ §



5. System Manageability

Network management is an increasingly important requirement in today's networked computer environment. Software-based management applications provide the ability to administer systems while the operating system is functioning in a normal power state (not in a pre-boot state or powered-down state). The Intel® System Management Bus (SMBus) Interface and the Network Controller - Sideband Interface (NC-SI) for the 82598 fills the management void that exists when the operating system is not running or fully functional.

This is accomplished by providing a mechanism by which manageability network traffic can be routed to and from a management controller. The 82598 provides two different and mutually exclusive bus interfaces for manageability traffic. The first is the Intel proprietary SMBus interface; several generations of Intel® Ethernet controllers have provided this same interface that operates at speeds of up to 1 MHz.

The second interface is NC-SI, which is a new industry standard interface created by the DMTF specifically for routing manageability traffic to and from a management controller. The NC-SI interface operates at 100 Mb/s full-duplex speeds.

5.1 Pass-Through (PT) Functionality

Pass-Through (PT) is the term used when referring to the process of sending and receiving Ethernet traffic over the sideband interface. The 82598 has the ability to route Ethernet traffic to the host operating system as well as the ability to send Ethernet traffic over the sideband interface to an external BMC.

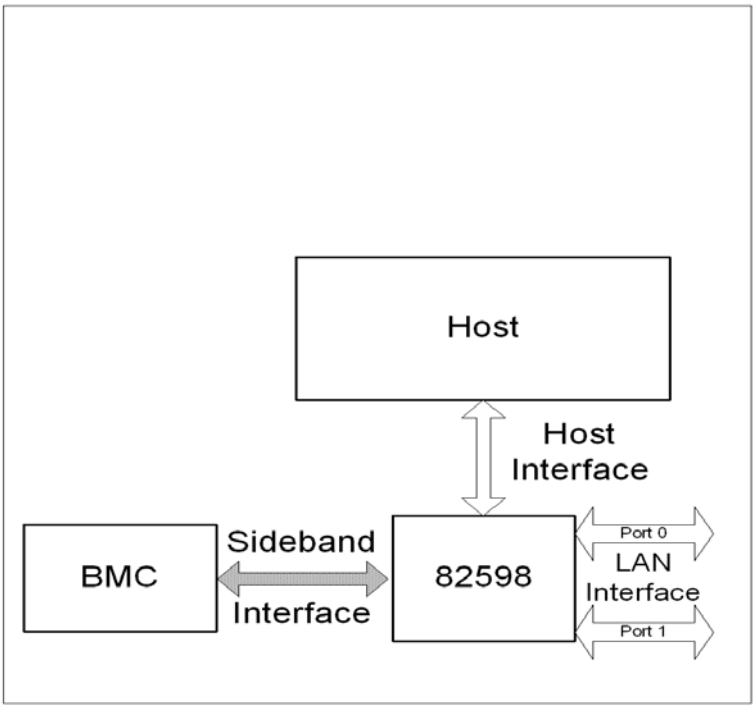


Figure 5-1. Sideband Interface

The sideband interface provides a mechanism by which the 82598 can be shared between the host and the BMC. By providing this sideband interface, the BMC can communicate with the LAN without requiring a dedicated Ethernet controller to do so.

The 82598 Eternet controller supports two sideband interfaces:

- SMBus
- NC-SI

The usable bandwidth for either direction is up to 400 Kb/s when using the SMBus interface and 100 Mb/s for the NC-SI interface.

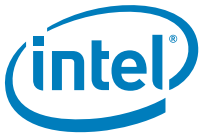
Note that only one mode of sideband can be active at any given time. This configuration is done via an EEPROM setting.

5.2 Components of a Sideband Interface

There are two components to a sideband interface:

- Physical Layer - The electrical layer that transfers data
- Logical Layer - the agreed upon protocol that is used for communications

The BMC and the 82598 must be in alignment for both of these components. For example, the NC-SI physical interface is based on the RMII interface. However, there are some differences at the physical level (detailed in the NC-SI specification) and the protocol layer is completely different.



5.3 SMBus Pass-Through Interface

SMBus is the system management bus defined by Intel® Corporation in 1995. It is used in personal computers and servers for low-speed system management communications. The SMBus interface is one of two pass-through interfaces available in the 82598 controller.

This section describes how the SMBus interface in the 82598 operates in pass-through mode.

5.3.1 General

The SMBus sideband interface includes the standard SMBus commands used for assigning a slave address and gathering device information as well as Intel proprietary commands used specifically for the pass-through interface.

5.3.2 Pass-Through Capabilities

This section details the specific manageability capabilities the 82598 provides while in SMBus mode.

The pass-through traffic is carried by the sideband interface as described in Section 5.1.

When operating in SMBus mode, in addition to exposing a communication channel to the LAN for the BMC, the 82598 provides the following manageability services to the BMC:

- ARP handling - The 82598 can be programmed to auto-ARP replying for ARP request packets to reduce the traffic over the BMC interconnect. See Section 5.3.3.2 for details.
- Teaming and fail-over - The 82598 can be configured to one of several teaming and fail-over configurations (See Section 5.3.11.1):
  - No-teaming - The 82598 dual LAN ports act independently of each other and no fail-over is provided by the 82598. The BMC is responsible for teaming and fail-over.
  - Teaming - The 82598 is configured to provide fail-over capabilities, such that manageability traffic is routed to an active port if any of the ports fail. Several modes of operation are provided.

Note: These services are not available in NC-SI mode.

5.3.2.1 Packet Filtering

Since the host OS and the BMC both use the 82598 Ethernet controller to send and receive Ethernet traffic, there needs to be a mechanism by which incoming Ethernet packets can be identified as those that should be sent to the BMC rather than the host OS.

In order to determine the types of traffic that is forwarded to the BMC over the sideband interface, the 82598 supports a manageability receive filtering mechanism. This mechanism is used to decide if a received packet should be forwarded to the BMC or to the host.

The following is a list of the filtering capabilities available for the SMBus interface with the 82598:

- RMCP/RMCP+ ports
- Flexible UDP/TCP port filters
- 128-byte flexible filters
- VLAN
- IPv4 address

- IPv6 address
- MAC address filters

Each of these are discussed in detail later in this section.

5.3.3 Pass-Through Multi-Port Modes

Pass-through configuration depends on the way the LAN ports are configured. If the LAN ports are configured as two different channels (non-teaming mode), then the 82598 is presented on the manageability link as two different devices. For example, via two different SMB addresses on which each device is connected to a different LAN port. In this mode (the same as in the LAN channels), there is no logical connection between the two devices. In this mode the fail-over between the two LAN ports is done by the external BMC (by sending/receiving packets through different devices). The status report to the BMC, ARP handling, DHCP, and other pass-through functionality are unique for each port.

When the LAN ports are configured to work as one LAN channel (teaming mode), the 82598 presents itself on the SMBus as one device (one SMB address). In this mode, the external BMC is not aware that there are two LAN ports. The 82598 decides how to route the packet that it receives from the LAN according to the fail-over algorithm. The status report to the BMC and other pass-through configuration are common to both ports.

5.3.3.1 Automatic Ethernet ARP Operation

Automatic Ethernet ARP parameters are loaded from the EEPROM when the 82598 is powered up or configured through the sideband management interface. The following parameters should be configured in order to enable ARP operation:

- ARP auto-reply enabled
- ARP IP address (to filter ARP packets)
- ARP MAC addresses (for ARP responses)

These are all configurable over the sideband interface using the advanced version of the Receive Enable command (see Section 5.3.10.1.3) or using the EEPROM.

When an ARP request packet is received on the wire, and ARP auto-reply is enabled, the 82598 checks the targeted IP address (after the packet has passed L2 checks and ARP checks). If the targeted IP matches the IP the 82598 was configured to, than it replies with an ARP response. The 82598 responds to ARP request targeted to the ARP IP address with the ARP MAC address configured to it. In case that there is no match, the 82598 silently discards the packets. If the 82598 is not configured to do auto-ARP response it forwards the ARP packets to the BMC.

When the external BMC uses the same IP and MAC address of the OS, the ARP operation should be coordinated with the OS operation. In this mode, this is the external BMC's responsibility and the 82598 stops/starts doing automatic ARP response as configured.

5.3.3.2 Manageability Receive Filtering

This section describes the manageability receive packet filtering flow when in SMBus mode. The description applies to a capability of each of the 82598 LAN ports. Packets that are received by the 82598 can be discarded, sent to the host memory, sent to the external BMC, or sent to both BMC and host memory.

5.3.3.2.1 Overview and General Structure

There are two modes of receive manageability filtering:

1. Receive Filtering - In this mode only certain types of packets are directed to the manageability block. The BMC should set the *RCV_TCO_EN* bit together with the specific packet type bits in the manageability filtering registers.

Note: The *RCV_ALL* bit must be cleared.

2. Receive All - all receive packets are routed to the BMC in this mode. It is enabled by setting the *RCV_TCO_EN* bit (which enables packets to be routed to the BMC) and the *RCV_ALL* bit (which routes all packets to the BMC) in the MANC register. RCV_ALL is only used for debug because it blocks all host traffic.

The default mode is that every packet that is directed to the BMC is not directed to host memory. The BMC enables the 82598 to direct certain manageability packets also to host memory by setting the *EN_MNG2HOST* bit in the MANC register. It then needs to configure the 82598 to send manageability packets to the host according to their type by setting the corresponding bits in the MANC2H register.

The BMC can control the types of packets that it receives by programming the receive manageability filters. Following is the list of filters that are accessible to the BMC:

Filters	Functionality	When Reset?
Filters Enable	General configuration of the manageability filters	Internal_Power_On_Reset and FW Reset
Manageability to Host	Enables routing of manageability packets to host	Internal_Power_On_Reset and FW Reset
Manageability Decision Filters [6:0]	Configuration of manageability decision filters	Internal_Power_On_Reset and FW Reset
MAC Address [3:0]	Four unicast MAC manageability addresses	Internal_Power_On_Reset
VLAN Filters [7:0]	Eight VLAN tag values	Internal_Power_On_Reset
UDP/TCP Port Filters [15:0]	16 destination port values	Internal_Power_On_Reset
Flexible 128 bytes TCO Filters [3:0]	Length values for four flex TCO filters	Internal_Power_On_Reset
IPv4 and IPv6 Address Filters [3:0]	IP address for manageability filtering	Internal_Power_On_Reset

All these filters are reset only on Internal_Power_On_Reset. Register filters that enable filters or functionality are also reset on FW reset. These registers can be loaded from the EEPROM following a reset.

The high-level structure of manageability filtering is done in three steps:

1. Packets are filtered by L2 criteria (MAC address, unicast/multicast/broadcast).
2. Packets are then filtered by VLAN if a VLAN tag is present.
3. Packets are filtered by the manageability filters (port, IP, flex, other), as configured in the decision filters.

Some general rules:

- Fragmented packets are passed to manageability but not parsed beyond the IP header.
- Packets with L2 errors (CRC, alignment, other) are never forwarded to manageability.

Note: If the BMC uses a dedicated MAC address/VLAN tag, it should take care not to use L3/L4 decision filtering on top of it. Otherwise all the packets with the manageability MAC address/ VLAN tag filtered out at L3/L4 are forwarded to the host.

The following sections describe each of these stages in detail.

5.3.3.2.2 L2 Layer Filtering

Figure 5-2 shows the manageability L2 filtering. A packet passes successfully through L2 filtering if any of the following conditions are met:

- It is a unicast packet and promiscuous unicast filtering is enabled from host.
- It is a multicast packet and promiscuous multicast filtering is enabled from host.
- It is a unicast packet and it matches one of the unicast MAC filters (host or manageability).
- It is a multicast packet and it matches one of the multicast filters.
- It is a broadcast packet. Note that in this case, the packet does not go through VLAN filtering (VLAN filtering is assumed to match).

Promiscuous unicast mode - Promiscuous unicast mode can be set/cleared only by the LAN device driver (not by the BMC), and it is usually used when the LAN device is used as a sniffer.

Promiscuous multicast mode - Promiscuous multicast is used in LAN devices that are used as a sniffer, and is controlled only by the LAN device driver. This bit can also be used by a BMC requiring forwarding of all multicasts.

Unicast filtering - The entire MAC address is checked against the 16 host unicast addresses and four management unicast addresses (if enabled). The 16 host unicast addresses are controlled by the LAN device driver (the BMC can not change them). The other four addresses are dedicated to management functions and are only accessed by the BMC.

The BMC configures manageability unicast filtering via update manageability receive filtering (see Section 5.3.10.1.6).

Multicast filtering - only 12 bits out of the packet's destination MAC address are compared against the multicast entries. These entries can be configured only by the LAN device driver and cannot be controlled by the BMC.

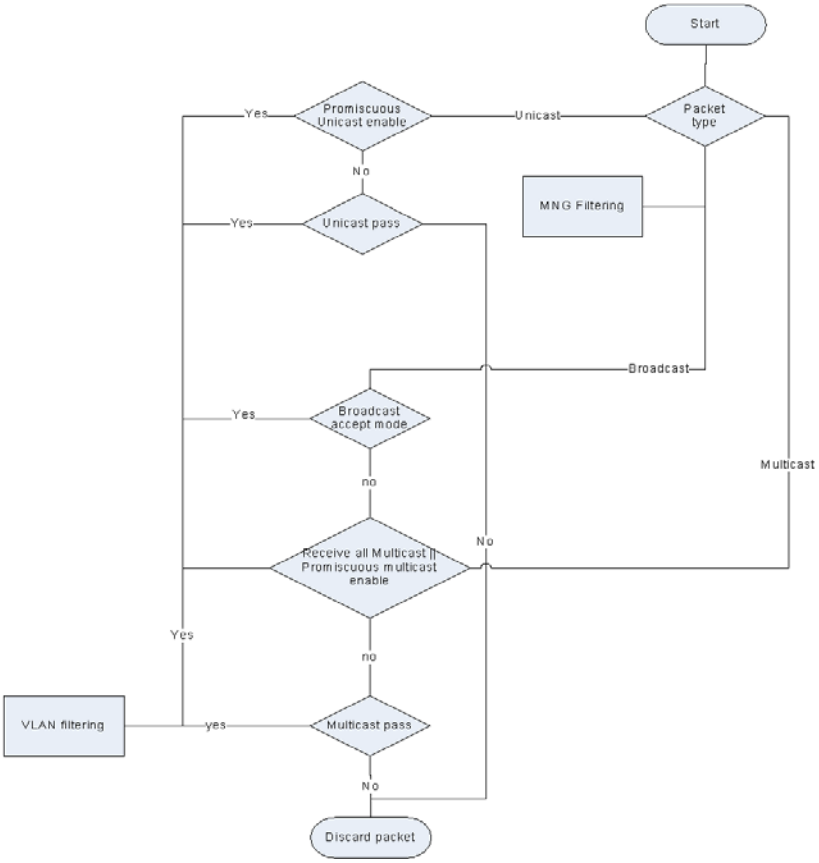


Figure 5-2. L2 Packet Filtering (Receive)

5.3.3.2.2.1 VLAN Filtering

Figure 5-3 shows the manageability VLAN filtering. A receive packet that passed L2 layer filtering successfully is then subjected to VLAN header filtering:

- If the packet does not have a VLAN header and the 82598 is not configured to receive only VLAN packets, it passes to the next filtering stage (manageability filtering).
- If the packet has a VLAN header and it passes a valid manageability VLAN filter and the 82598 is configured to receive VLAN packets, it passes to the next filtering stage (manageability filtering).
- If the packet has a VLAN header, the 82598 manageability is configured to receive VLAN packets, and it matches an enabled host VLAN filter, the packet is forwarded to the next filtering stage (manageability filtering).
- Otherwise, the packet is dropped.

The BMC configures the 82598 with up to eight different manageability VLAN IDs (VIDs) via the Management VLAN Tag Filters (see Section 5.3.3.2.2.1).

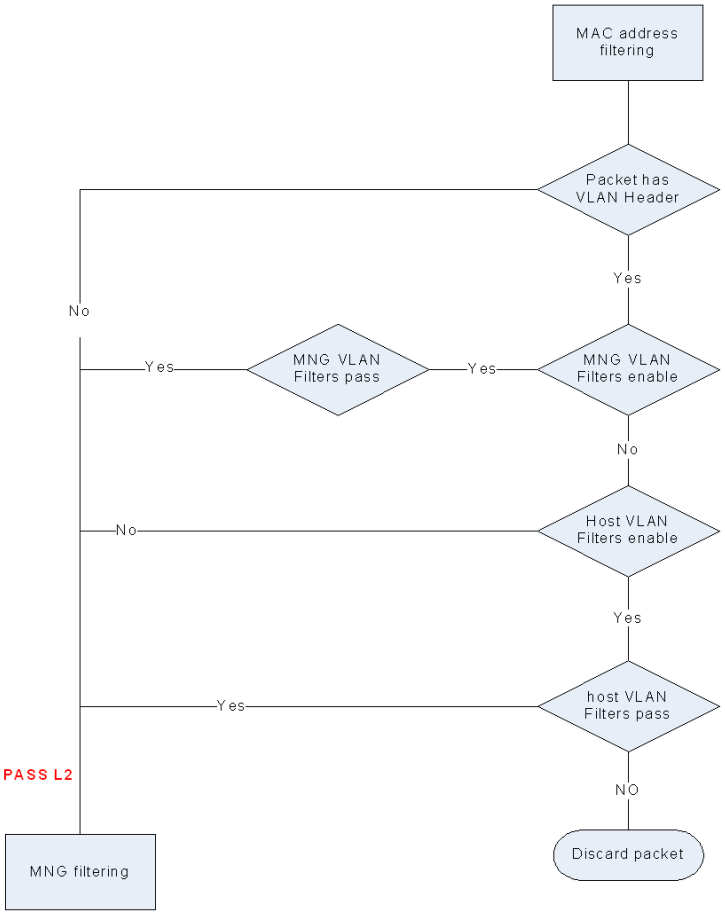


Figure 5-3. Manageability VLAN Header Filtering (Receive)

5.3.3.2.3 Manageability Decision Filtering

The manageability decision filtering stage combines some of the checks done at the previous stages with additional L3/L4 checks into a final decision whether to route a packet to the BMC. This section describes the additional filters done at layers L3 &L4, followed by the final filtering rules.

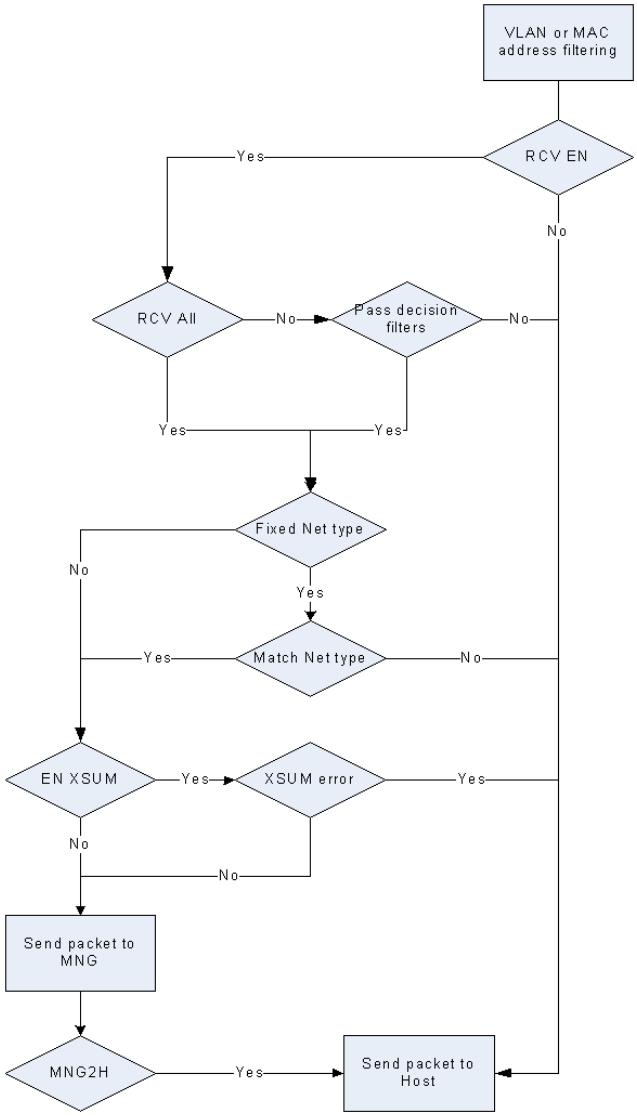


Figure 5-4. Manageability Filtering (Receive)

5.3.3.2.3.1 L3 & L4 Filters

ARP Filtering

The 82598 supports filtering of both ARP request packets (initiated externally) and ARP responses (to requests initiated by the BMC or the 82598).

Neighbor Discovery Filtering

The 82598 supports filtering of neighbor discovery packets. Neighbor discovery uses the IPv6 destination address filters defined in the MIPAF registers (for instance, all enabled IPv6 addresses are matched for neighbor discovery).

Port 0x298/0x26F Filtering

The 82598 supports filtering by fixed destination ports numbers, port 0x26F and port 0x298.

Flex Port Filtering

The 82598 implements 16 flex destination port filters. The 82598 directs packets that their L4 destination port and matches the value of the respective word in the MFUTP registers. The BMC must insure that only valid entries are enabled in the decision filters.

Flex TCO Filters

The 82598 provides four flex TCO filters. Each filter looks for a pattern match within the first 128 bytes of the packet. The BMC configures the pattern to match into the FTFT table, and the length in the last two entries of the FFLT table. The BMC must insure that only valid entries are enabled in the decision filters.

IP Address Filtering

The 82598 supports filtering by IP address through IPv4 and IPv6 address filters, dedicated to manageability. Two modes are possible, depending on the value of the MANC.EN_IPv4_FILTER bit:

- EN_IPv4_FILTER = 0b: The 82598 provides four IPv6 address filters.
- EN_IPv4_FILTER = 1b: The 82598 provides three IPv6 address filters and 4 IPv4 address filters.
- The MFVAL register indicates which of the IP address filters are valid (for instance, contains a valid entry and should be used for comparison).

Checksum Filter

If bit MANC.EN_XSUM_FILTER is set, the 82598 directs packets to the BMC only if they pass (if exists) L3/L4 checksum, in addition to matching the previously mentioned filters. Enabling this filter makes it so the BMC does not need to do the L3/L4 checksum verification, as a packet that fails this filter will never be sent to the BMC.

To enable the checksum filter, the BMC uses the Update Management Receive Filter Parameters command (see Section 5.3.10.1.6) with the parameter of 0x1 to configure the MANC register, setting the *EN_XSUM_FILTER* bit (bit 23).

5.3.3.2.4 Manageability Decision Filters

The manageability decision filters are a set of eight filters with the same structure. The filtering rule for each decision filter is programmed by the BMC and defines which of the filters (L2, VLAN, manageability) participate in the decision. A packet that passes at least one rule is directed to manageability and possibly to the host.

The inputs to each decision filter are:

- Packet passed a valid management L2 unicast address filter
- Packet is a broadcast packet
- Packet has a VLAN header and it passed a valid manageability VLAN filter
- Packet matched one of the valid IPv4 or IPv6 manageability address filters
- Packet passed ARP filtering (request or response)
- Packet passed neighbor discovery filtering
- Packet passed 0x298/0x26F port filter
- Packet passed a valid flex port filter
- Packet passed a valid flex TCO filter

- Packet is multicast

The structure of each of the decision filters is shown in Figure 3 4. A boxed number indicates that the input is conditioned on a mask bit defined in the MDEF register for this rule. The decision filter rules are as follows:

- At least one bit must be set in a MDEF register. If all bits are cleared (such as MDEF=0x0000), then the decision filter is disabled and ignored.
- All enabled AND filters must match for the decision filter to match. An AND filter not enabled in the MDEF register is ignored.
- If no OR filter is enabled in the MDEF register, the OR filters are ignored in the decision (such as the filter might still match).
- If at least one OR filter is enabled in the MDEF register, then at least one of the enabled OR filters must match for the decision filter to match.

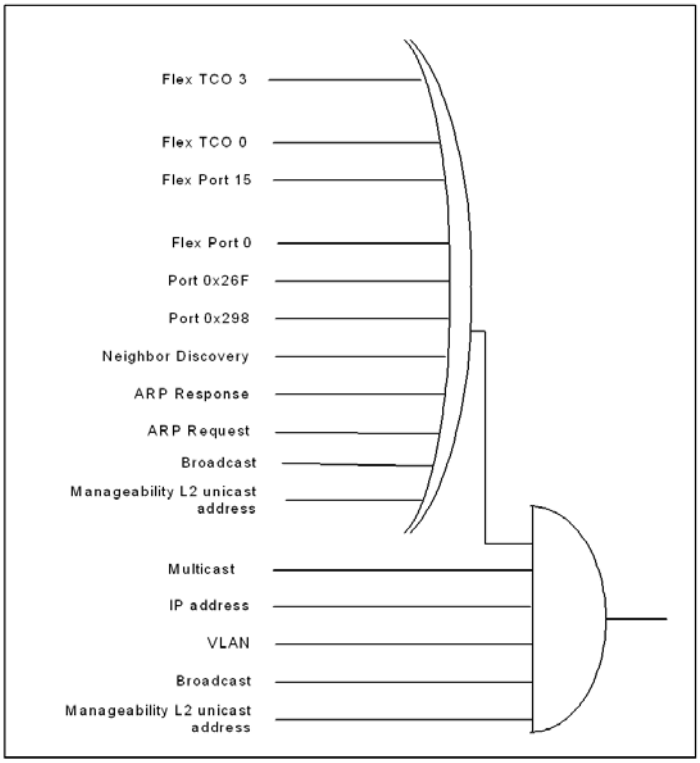


Figure 5-5. Manageability Decision Filters

A decision filter (for any of the seven filters) defines which of the inputs is enabled as part of the rule. The BMC programs a 32-bit rule with the settings as listed in Table 5-1. A set bit enables its corresponding filter to participate in the filtering decision.

Table 5-1. Assignment of Decision Filters

Filter	And / OR Input	Mask Bits in MDEF[7:0]
L2 unicast address	AND	0
Broadcast	AND	1
Manageability VLAN	AND	2
IP address	AND	3
L2 unicast address	OR	4
Broadcast	OR	5
Multicast	AND	6
ARP request	OR	7
ARP response	OR	8
Neighbor discovery	OR	9
Port 0x298	OR	10
Port 0x26F	OR	11
Flex port 15:0	OR	27:12
Flex TCO 3:0	OR	31:28

The default mode is that every packet that is directed to the BMC is not directed to host memory. The BMC can also enable the 82598 to direct certain manageability packets to host memory by setting the *EN_MNG2HOST bit* in the MANC register and then configuring the 82598 to send manageability packets to the host according to their type by setting the corresponding bits in the manageability to host filter (one bit per each of the seven decision rules).

The Mng2Host register has the following structure:

Bits	Description	Default
0	Decision Filter 0	Determines if packets that have passed decision filter 0 are also forwarded to the host OS.
1	Decision Filter 1	Determines if packets that have passed decision filter 1 are also forwarded to the host OS.
2	Decision Filter 2	Determines if packets that have passed decision filter 2 are also forwarded to the host OS.
3	Decision Filter 3	Determines if packets that have passed decision filter 3 are also forwarded to the host OS.

4	Decision Filter 4	Determines if packets that have passed decision filter 4 are also forwarded to the host OS.
5	Unicast and Mixed	Determines if broadcast packets are also forwarded to the host OS.
6	Global Multicast	Determines if unicast packets are also forwarded to the host OS.
7	Broadcast	Determines if multicast packets are also forwarded to the host OS.

All manageability filters are controlled by the BMC only and not by the LAN device driver.

The BMC enables these filters by issuing the Update Management Receive Filter Parameters command (see Section 5.3.10.1.6) with the parameter of 0x60.

5.3.4 SMBus Transactions

This section gives a brief overview of the SMBus protocol.

Following is an example for a format of a typical SMBus transaction:

1	7	1	1	8	1	8	1	1
S	Slave Address	Wr	A	Command	A	PEC	A	P
	1100 001	0	0	0000 0010	0	[Data Dependent]	0	

The top row of the table identifies the bit length of the field in a decimal bit count. The middle row (bordered) identifies the name of the fields used in the transaction. The last row appears only with some transactions, and lists the value expected for the corresponding field. This value can be either hexadecimal or binary.

The shaded fields are fields that are driven by the slave of the transaction. The un-shaded fields are fields that are driven by the master of the transaction. The SMBus controller is a master for some transactions and a slave for others. The differences are identified in this section.

Shorthand field names are listed in Table 5-2 and are fully defined in the SMBus specification:

Table 5-2. Shorthand Field Name

Field Name	Definition
S	SMBus START Symbol
P	SMBus STOP Symbol
PEC	Packet Error Code
A	ACK (Acknowledge)
N	NACK (Not Acknowledge)
Rd	Read Operation (Read Value = 1b)
Wr	Write Operation (Write Value = 0b)

5.3.4.1 SMBus Addressing

The SMBus addresses that the 82598 responds to depends on the LAN mode (teaming/non-teaming). If the LAN is in teaming mode (fail-over), the 82598 is presented over the SMBus as one device and has one SMBus address. While in non-teaming mode in the LAN ports, the SMBus is presented as two SMBus devices on the SMBus (two SMBus addresses). In dual-address mode, all pass-through functionality is duplicated on the SMBus address, where each SMBus address is connected to a different LAN port. Note that it is not allowed to configure both ports to the same SMBus address. When a LAN function is disabled, the corresponding SMBus address is not presented to the BMC (see Section 5.3.11.1).

The SMBus addressing mode is defined through the SMBus *Addressing Mode* bit in the EEPROM. The SMBus addresses are set in SMBus address 0 and SMBus address 1 in the EEPROM. Note that if single-address mode is set, only the SMBus address 0 field is valid.

The SMBus addresses (enabled from the EEPROM) can be re-assigned using the SMBus ARP protocol.

In addition to the SMBus address values, all parameters of the SMBus (SMBus channel selection, address mode, and address enable) can be set only through EEPROM configuration. Note that the EEPROM is read at 82598 power up and resets.

All SMBus addresses should be in Network Byte Order (NBO); MSB first.

5.3.4.2 SMBus ARP Functionality

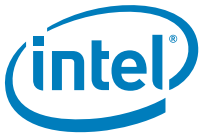
The 82598 supports the SMBus ARP protocol as defined in the SMBus 2.0 specification. The 82598 is a persistent slave address device so its SMBus address is valid after power-up and loaded from the EEPROM. The 82598 supports all SMBus ARP commands defined in the SMBus specification both general and directed.

Note: The SMBus ARP capability can be disabled through the EEPROM.

5.3.4.2.1 SMBus ARP Flow

SMBus ARP flow is based on the status of two flags:

- AV (Address Valid): This flag is set when the 82598 has a valid SMBus address.
- AR (Address Resolved): This flag is set when the 82598 SMBus address is resolved (SMBus address was assigned by the SMBus ARP process).



Note: These flags are internal 82598 flags and are not exposed to external SMBus devices.

Since the 82598 is a Persistent SMBus Address (PSA) device, the AV flag is always set, while the AR flag is cleared after power up until the SMBus ARP process completes. Since AV is always set, the 82598 always has a valid SMBus address.

When the SMBus master needs to start an SMBus ARP process, it resets (in terms of ARP functionality) all devices on the SMBus by issuing either Prepare to ARP or Reset Device commands. When the 82598 accepts one of these commands, it clears its AR flag (if set from previous SMBus ARP process), but not its AV flag (The current SMBus address remains valid until the end of the SMBus ARP process).

Clearing the AR flag means that the 82598 responds to the following SMBus ARP transactions that are issued by the master. The SMBus master issues a Get UDID command (general or directed) to identify the devices on the SMBus. The 82598 always responds to the Directed command and to the General command only if its AR flag is not set. After the Get UDID, The master assigns the 82598 SMBus address by issuing an Assign Address command. The 82598 checks whether the UDID matches its own UDID and if it matches, it switches its SMBus address to the address assigned by the command (byte 17). After accepting the Assign Address command, the AR flag is set and from this point (as long as the AR flag is set), the 82598 does not respond to the Get UDID General command. Note that all other commands are processed even if the AR flag is set. The 82598 stores the SMBus address that was assigned in the SMBus ARP process in the EEPROM, so at the next power up, it returns to its assigned SMBus address.

SMBus ARP Flow shows the 82598 SMBus ARP flow.

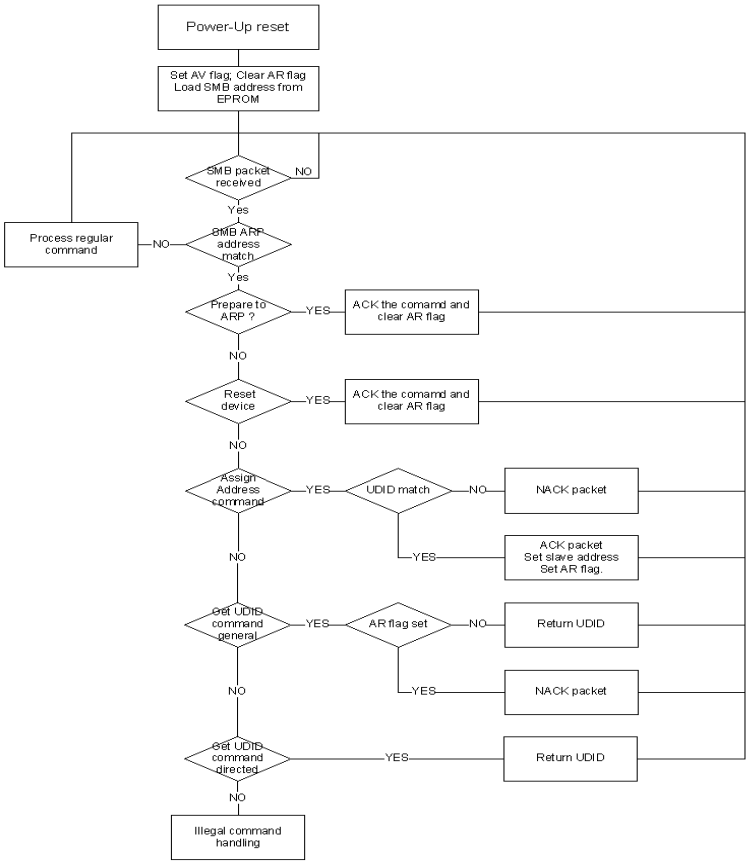


Figure 5-6. SMBus ARP Flow

5.3.4.2.2 SMBus ARP UDID Content

The UDID provides a mechanism to isolate each device for the purpose of address assignment. Each device has a unique identifier. The 128-bit number is comprised of the following fields:

1 Byte	1 Byte	2 Bytes	2 Bytes	2 Bytes	2 Bytes	2 Bytes	4 Bytes
Device Capabilities	Version/ Revision	Vendor ID	Device ID	Interface	Subsystem Vendor ID	Subsystem Device ID	Vendor Specific ID
See Below	See Below	0x8086	0x10AA	0x0004	0x0000	0x0000	See Below
MSB							LSB

Where:

Vendor ID:	The device manufacturer's ID as assigned by the SBS Implementers' Forum or the PCI SIG. Constant value: 0x8086
Device ID:	The device ID as assigned by the device manufacturer (identified by the Vendor ID field). Constant value: 0x10AA
Interface:	Identifies the protocol layer interfaces supported over the SMBus connection by the device. In this case, SMBus Version 2.0 Constant value: 0x0004
Subsystem Fields:	These fields are not supported and return zeros.

Device Capabilities: Dynamic and Persistent Address, *PEC Support* bit:

7	6	5	4	3	2	1	0
Address Type		Reserved (0)	Reserved (0)	Reserved (0)	Reserved (0)	Reserved (0)	PEC Supported
0b	1b	0b	0b	0b	0b	0b	0b
MSB							LSB

Version/Revision: UDID Version 1, Silicon Revision:

7	6	5	4	3	2	1	0
Reserved (0)	Reserved (0)	UDID Version			Silicon Revision ID		
0b	0b	001b			See Below		
MSB							LSB

Silicon Revision ID:

Silicon Version	Revision ID
A0	000b
A1	001b
A2	010b

Vendor Specific ID: Four LSB bytes of the device Ethernet MAC Address. The device Ethernet address is taken from words 0x2:0x0 in the EEPROM. Note that in the 82598 there are two MAC addresses (one for each port). Bit 0 of the port 1 MAC address has the inverted value of bit 0 from the EEPROM.

1 Byte	1 Byte	1 Byte	1 Byte
MAC Address, Byte 3	MAC Address, Byte 2	MAC Address, Byte 1	MAC Address, Byte 0
MSB			LSB

5.3.4.2.3 SMBus ARP in Dual/Single Mode

The 82598 operates in either single SMBus address mode or in dual SMBus address mode. These modes reflect its SMBus ARP behavior.

While operating in single mode, the 82598 presents itself on the SMBus as one device and only responds to SMBus ARP as one device. In this case, the 82598's SMBus address is SMBus address 0 as defined in the EEPROM SMBus ARP address word. The 82598 has only one AR and AV flag. The vendor ID, the MAC address of the LAN's port, is taken from the port 0 address.

In dual mode, the 82598 responds as two SMBus devices having two sets of AR/AV flags (one for each port). The 82598 responds twice to the SMBus ARP master, once each for each port. Both SMBus addresses are taken from the SMBus ARP address word of the EEPROM. Note that the Unique Device Identifier (UDID) is different between the two ports in the version ID field, which represents the MAC address and is different between the two ports. It is recommended that the 82598 first respond as port 0, and only when the address is assigned, to start responding as port 1 to the Get UDID command.

5.3.4.3 Concurrent SMBus Transactions

In single-address mode, concurrent SMBus transactions (receive, transmit and configuration read/write) are allowed without limitation. Transmit fragments can be sent between receive fragments and configuration Read/Write commands can also issue between receive and transmit fragments.

In dual-address mode, the same rules apply to concurrent traffic between the two addresses supported by the 82598.

Note: Packets can only be transmitted from one port/device at a given time. As a result, the BMC must finish sent packets (send a last fragment command) from one port before starting the transmission for the other port.

5.3.5 SMBus Notification Methods

The 82598 supports three methods of notifying the BMC that it has information that needs to be read by the BMC:

- SMBus alert
- Asynchronous notify
- Direct receive

The notification method that is used by the 82598 can be configured from the SMBus using the Receive Enable command. This default method is set by the EEPROM in the pass-through init field.

The following events cause the 82598 to send a notification event to the BMC:

- Receiving a LAN packet that is designated to the BMC.

- Receiving a Request Status command from the BMC initiates a status response.
- Status change has occurred and the 82598 is configured to notify the external BMC at one of the status changes.
- Change in any in the Status Data 1 bits of the Read Status command.

There can be cases where the BMC is hung and therefore not responding to the SMBus notification. The 82598 has a time-out value (defined in the EEPROM) to avoid hanging while waiting for the notification response. If the BMC does not respond until the time out expires, the notification is de-asserted and all pending data is silently discarded.

Note that the SMBus notification time-out value can only be set in the EEPROM, the BMC cannot modify this value.

5.3.5.1 SMBus Alert and Alert Response Method

The SMBus Alert# (SMBALERT_N) signal is an additional SMBus signal that acts as an asynchronous interrupt signal to an external SMBus master. The 82598 asserts this signal each time it has a message that it needs the BMC to read and if the chosen notification method is the SMBus alert method. Note that the SMBus alert method is an open-drain signal which means that other devices besides the 82598 can be connected on the same alert pin. As a result, the BMC needs a mechanism to distinguish between the alert sources.

The BMC can respond to the alert, by issuing an ARA Cycle command, to detect the alert source device. The 82598 responds to the ARA cycle with its own SMBus slave address (if it was the SMBus alert source) and de-asserts the alert when the ARA cycle is completes. Following the ARA cycle, the BMC issues a read command to retrieve the 82598 message.

Some BMCs do not implement the ARA cycle transaction. These BMCs respond to an alert by issuing a Read command to the 82598 (0xC0/0xD0 or 0xDE). The 82598 always responds to a Read command, even if it is not the source of the notification. The default response is a status transaction. If the 82598 is the source of the SMBus Alert, it replies the read transaction and then de-asserts the alert after the command byte of the read transaction.

Note: In SMBus Alert mode, the SMBALERT_N pin is used for notification. In dual-address mode, both devices generate alerts on events that are independent of each other.

The ARA cycle is a SMBus Receive Byte transaction to SMBus address 0x18. Note that the ARA transaction does not support PEC. The alert response address transaction format is shown in Figure 5-7.

1	7	1	1	8	1	1
S	Alert Response Address	Rd	A	Slave Device Address	A	P
	0001 100	0	0		1	

Figure 5-7. SMBus ARA Cycle Format

5.3.5.2 Asynchronous Notify Method

When configured using the asynchronous notify method, the 82598 acts as a SMBus master and notifies the BMC by issuing a modified form of the write word transaction. The asynchronous notify transaction SMBus address and data payload is configured using the Receive Enable command or using the EEPROM defaults. Note that the asynchronous notify is not protected by a PEC byte.

1	7	1	1	7	1	1	
S	Target Address	Wr	A	Sending Device Address		A	. . .
	BMC Slave Address	0	0	MNG Slave SMBus Address	0	0	

8	1	8	1	1
Data Byte Low	A	Data Byte High	A	P
Interface	0	Alert Value	0	

Figure 5-8. Asynchronous Notify Command Format

The target address and data byte low/high is taken from the Receive Enable command or EEPROM configuration.

5.3.5.3 Direct Receive Method

If configured, the 82598 has the capability to send a message it needs to transfer to the external BMC as a master over the SMBus instead of alerting the BMC and waiting for it to read the message.

The message format is shown below. Note that the command that is used is the same command that is used by the external BMC in the Block Read command. The opcode that the 82598 puts in the data is also the same as it put in the Block Read command of the same functionality. The rules for the *F* and *L* flags (bits) are also the same as in the Block Read command.

1	7	1	1	1	1	6	1	
S	Target Address	Wr	A	F	L	Command	A	. . .
	BMC Slave Address	0	0	First Flag	Last Flag	Receive TCO Command 01 0000b	0	

8	1	8	1		1	8	1	1
Byte Count	A	Data Byte 1	A	. . .	A	Data Byte N	A	P
N	0		0		0		0	

Figure 5-9. Direct Receive Transaction Format

5.3.6 1 MHz SMBus Support

SMBus specification defines the maximum frequency of the SMBus as 100 KHz. The 82598 SMBus can be activated up to frequency of 1 MHz. When operating at 1 MHz, few of the SMBus specification parameters are violated. The regular SMBus can be activated in two modes: slow, which meets the SMBus specification requirements (can be activated up to 400 KHz without violating hold and setup time) and fast, which can be operated up to 1 MHz, but does not meet the SMBus specification.

This configuration is only available via an EEPROM setting.

The EEPROM Pass-Through SMBus Connection field defines the SMBus mode (slow SMBus/fast SMBus). The slow SMBus DC parameters are defined in the SMBus 2.0 specification.

5.3.7 Receive TCO Flow

The 82598 is used as a channel for receiving packets from the network link and passing them to the external BMC. The BMC configures the 82598 to pass these specific packets to the BMC. Once a full packet is received from the link and identified as a manageability packet that should be transferred to the BMC, the 82598 starts the receive TCO flow to the BMC.

The 82598 uses the SMBus notification method to notify the BMC that it has data to deliver. Since the packet size might be larger than the maximum SMBus fragment size, the packet is divided into fragments, where the 82598 uses the maximum fragment size allowed in each fragment (configured via the EEPROM). The last fragment of the packet transfer is always the status of the packet. As a result, the packet is transferred in at least two fragments. The data of the packet is transferred as part of the receive TCO LAN packet transaction.

When SMBus alert is selected as the BMC notification method, the 82598 notifies the BMC on each fragment of a multi fragment packet. When asynchronous notify is selected as the BMC notification method, the 82598 notifies the BMC only on the first fragment of a received packet. It is the BMC's responsibility to read the full packet including all the fragments.

Any timeout on the SMBus notification results in discarding the entire packet. Any NACK by the BMC causes the fragment to be re-transmitted to the BMC on the next Receive Packet command.

The maximum size of the received packet is limited by the 82598 hardware to 1536 bytes. Packets larger then 1536 bytes are silently discarded. Any packet smaller than 1536 bytes is processed by the 82598.

5.3.8 Transmit TCO Flow

The 82598 is used as the channel for transmitting packets from the external BMC to the network link. The network packet is transferred from the BMC over the SMBus and then, when fully received by the 82598, is transmitted over the network link.

In dual-address mode, each SMBus address is connected to a different LAN port. When a packet is received during a SMBus transaction using SMBus address #0, it is transmitted to the network using LAN port #0 and is transmitted through LAN port #1, if received on SMBus address #1. In single address mode, the transmitted port is chosen according to the fail-over algorithm.

The 82598 supports packets up to an Ethernet packet length of 1536 bytes. Since SMBus transactions can only be up to 240 bytes in length, packets might need to be transferred over the SMBus in more than one fragment. This is achieved using the *F* and *L* bits in the command number of the transmit TCO packet Block Write command. When the *F* bit is set, it is the first fragment of the packet. When the *L* bit is set, it is the last fragment of the packet. When both bits are set, the entire packet is in one fragment. The packet is sent over the network link, only after all its fragments are received correctly over the SMBus. The maximum SMBus fragment size is defined within the EEPROM and cannot be changed by the BMC.

If the packet sent by the BMC is larger than 1536 bytes, than the packet is silently discarded by the 82598. The minimum packet length defined by the 802.3 spec is 64 bytes. The 82598 pads packets that are less than 64 bytes to meet the specification requirements (there is no need for the external BMC to pad packets less than 64 bytes). If the packet sent by the BMC is larger than 1536 bytes the 82598 silently discards the packet.

The 82598 calculates the L2 CRC on the transmitted packet and adds its four bytes at the end of the packet. Any other packet field (such as XSUM) must be calculated and inserted by the BMC (the 82598 does not change any field in the transmitted packet, other than adding padding and CRC bytes).

If the network link is down when the 82598 has received the last fragment of the packet from the BMC, it silently discards the packet. Note that any link down event during the transfer of any packet over the SMBus does not stop the operation since the 82598 waits for the last fragment to end to see whether the network link is up again.

5.3.8.1 Transmit Errors in Sequence Handling

Once a packet is transferred over the SMBus from the BMC to the 82598, the *F* and *L* flags should follow specific rules. The *F* flag defines that this is the first fragment of the packet; the *L* flag defines that the transaction contains the last fragment of the packet.

Flag options during transmit packet transactions lists the different flag options in transmit packet transactions:

Table 5-3. Flag Options During Transmit Packet Transactions

Previous	Current	Action/Notes
Last	First	Accept both.
Last	Not First	Error for the current transaction. Current transaction is discarded and an abort status is asserted.
Not Last	First	Error in previous transaction. Previous transaction (until previous First) is discarded. Current packet is processed. No abort status is asserted.
Not Last	Not First	Process the current transaction.

Note: Since every other Block Write command in TCO protocol has both *F* and *L* flags off, they cause flushing any pending transmit fragments that were previously received. When running the TCO transmit flow, no other Block Write transactions are allowed in between the fragments.

5.3.8.2 TCO Command Aborted Flow

The 82598 indicates to the BMC an error or an abort condition by setting the TCO Abort bit in the general status. The 82598 might also be configured to send a notification to the BMC (see Section 5.3.10.1.3.3).

Following is a list of possible error and abort conditions:

- Any error in the SMBus protocol (NACK, SMBus timeouts, etc.).
- Any error in compatibility between required protocols to specific functionality (for example, RX Enable command with a byte count not equal to 1/14, as defined in the command specification).
- If the 82598 does not have space to store the transmitted packet from the BMC (in its internal buffer space) before sending it to the link, the packet is discarded and the external BMC is notified via the *Abort* bit.
- Error in the *F/L* bit sequence during multi-fragment transactions.
- An internal reset to the 82598's firmware.

5.3.9 SMBus ARP Transactions

Note: All SMBus ARP transactions include the PEC byte.

5.3.9.1 Prepare to ARP

This command clears the *Address Resolved* flag (set to false). It does not affect the status or validity of the dynamic SMBus address and is used to inform all devices that the ARP master is starting the ARP process:

1	7	1	1	8	1	8	1	1
S	Slave Address	Wr	A	Command	A	PEC	A	P
	1100 001	0	0	0000 0001	0	[Data Dependent Value]	0	

5.3.9.2 Reset Device (General)

This command clears the *Address Resolved* flag (set to false). It does not affect the status or validity of the dynamic SMBus address.

1	7	1	1	8	1	8	1	1
S	Slave Address	Wr	A	Command	A	PEC	A	P
	1100 001	0	0	0000 0010	0	[Data Dependent Value]	0	

5.3.9.3 Reset Device (Directed)

The Command field is NACKed if bits 7:1 do not match the current 82598 SMBus address. This command clears the *Address Resolved* flag (set to false) and does not affect the status or validity of the dynamic SMBus address.

1	7	1	1	8	1	8	1	1
S	Slave Address	Wr	A	Command	A	PEC	A	P
	1100 001	0	0	Targeted Slave Address | 0	0	[Data Dependent Value]	0	

5.3.9.4 Assign Address

This command assigns the 82598 SMBus address. The address and command bytes are always acknowledged.

The transaction is aborted (NACKed) immediately if any of the UDID bytes is different from the 82598 UDID bytes. If successful, the manageability system internally updates the SMBus address. This command also sets the *Address Resolved* flag (set to true).

1	7	1	1	8	1	8	1	
S	Slave Address	Wr	A	Command	A	Byte Count	A	•••
	1100 001	0	0	0000 0100	0	0001 0001	0	

8	1	8	1	8	1	8	1	
Data 1	A	Data 2	A	Data 3	A	Data 4	A	•••
UDID Byte 15 (MSB)	0	UDID Byte 14	0	UDID Byte 13	0	UDID Byte 12	0	

8	1	8	1	8	1	8	1	
Data 5	A	Data 6	A	Data 7	A	Data 8	A	•••
UDID Byte 11	0	UDID Byte 10	0	UDID Byte 9	0	UDID Byte 8	0	

8	1	8	1	8	1	
Data 9	A	Data 10	A	Data 11	A	•••
UDID Byte 7	0	UDID Byte 6	0	UDID Byte 5	0	

8	1	8	1	8	1	8	1	
Data 12	A	Data 13	A	Data 14	A	Data 15	A	•••
UDID Byte 4	0	UDID Byte 3	0	UDID Byte 2	0	UDID Byte 1	0	

8	1	8	1	8	1	1
Data 16	A	Data 17	A	PEC	A	P
UDID Byte 0 (LSB)	0	Assigned Address	0	[Data Dependent Value]	0	

5.3.9.5 Get UDID (General and Directed)

The general get UDID SMBus transaction supports a constant command value of 0x03 and in directed, supports a Dynamic command value equal to the dynamic SMBus address.

If the SMBus address has been resolved (*Address Resolved* flag set to true), the manageability system does not acknowledge (NACK) this transaction. If its a General command, the manageability system always acknowledges (ACKs) as a directed transaction.

This command does not affect the status or validity of the dynamic SMBus address or the *Address Resolved* flag.

S	Slave Address	Wr	A	Command	A	S	•••
	1100 001	0	0	See Below	0		

7	1	1	8	1	

Slave Address	Rd	A	Byte Count	A	•••
1100 001	1	0	0001 0001	0	

8	1	8	1	8	1	8	1	
Data 1	A	Data 2	A	Data 3	A	Data 4	A	•••
UDID Byte 15 (MSB)	0	UDID Byte 14	0	UDID Byte 13	0	UDID Byte 12	0	

8	1	8	1	8	1	8	1	
Data 5	A	Data 6	A	Data 7	A	Data 8	A	•••
UDID Byte 11	0	UDID Byte 10	0	UDID Byte 9	0	UDID Byte 8	0	

8	1	8	1	8	1	
Data 9	A	Data 10	A	Data 11	A	•••
UDID Byte 7	0	UDID Byte 6	0	UDID Byte 5	0	

8	1	8	1	8	1	8	1	
Data 12	A	Data 13	A	Data 14	A	Data 15	A	•••
UDID Byte 4	0	UDID Byte 3	0	UDID Byte 2	0	UDID Byte 1	0	

8	1	8	1	8	1	1
Data 16	A	Data 17	A	PEC	~Ã	P
UDID Byte 0 (LSB)	0	Device Slave Address	0	[Data Dependent Value]	1	

The Get UDID command depends on whether or not this is a Directed or General command.

The General Get UDID SMBus transaction supports a constant command value of 0x03.

The Directed Get UDID SMBus transaction supports a Dynamic command value equal to the dynamic SMBus address with the LSB bit set.

Note: Bit 0 (LSB) of Data byte 17 is always 1b.

5.3.10 SMBus Pass-Through Transactions

This section details all of the commands (both read and write) that the 82598 SMBus interface supports for pass-through.

5.3.10.1 Write Transactions

This section details the commands that the BMC can send to the 82598 over the SMBus interface. SMBus write transactions table lists the different SMBus write transactions supported by the 82598.

TCO Command	Transaction	Command	Fragmentation	Section
Transmit Packet	Block Write	First: 0x84 Middle: 0x04 Last: 0x44	Multiple	5.3.10.1.1
Transmit Packet	Block Write	Single: 0xC4	Single	5.3.10.1.1
Request Status	Block Write	Single: 0xDD	Single	5.3.10.1.2
Receive Enable	Block Write	Single: 0xCA	Single	5.3.10.1.3
Force TCO	Block Write	Single: 0xCF	Single	5.3.10.1.4
Management Control	Block Write	Single: 0xC1	Single	5.3.10.1.5
Update MNG RCV Filter Parameters	Block Write	Single: 0xCC	Single	5.3.10.1.6

5.3.10.1.1 Transmit Packet Command

Note: If the overall packet length is greater than 1536 bytes, the packet is silently discarded by the 82598.

5.3.10.1.2 Request Status Command

An external BMC can initiate a request to read the 82598 manageability status by sending a Request Status command. When received, the 82598 initiates a notification to an external BMC (when status is ready), after which, an external BMC is able to read the status by issuing this command. The format is as follows:

Function	Command	Byte Count	Data 1
Request Status	0xDD	1	0

5.3.10.1.3 Receive Enable Command

The Receive Enable command is a single fragment command used to configure the 82598. This command has two formats: short, 1-byte legacy format (providing backward compatibility with previous components) and long, 14-byte advanced format (allowing greater configuration capabilities). The Receive Enable command format is as follows:

Function	CMD	Byte Count	Data 1	Data 2	...	Data 7	Data 8	...	Data 11	Data 12	Data 13	Data 14
Legacy Receive Enable	0xCA	1	Receive Control Byte	-	...	-	-	...	-	-	-	-
Advanced Receive Enable		14 (0x0E)		MAC Addr LSB		MAC Addr MSB	IP Addr LSB		IP Addr MSB	BMC SMBus Addr	I/F Data Byte	Alert Value Byte

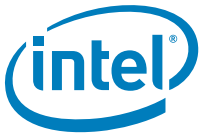


Table 5-4. Receive Control Byte (Data Byte)

Field	Bit(s)	Description
RCV_EN	0	Receive TCO Enable. 0b: Disable receive TCO packets. 1b: Enable Receive TCO packets. Setting this bit enables all manageability receive filtering operations. Enabling specific filters is done via the EEPROM or through special configuration commands. **Note:** When the RCV_EN bit is cleared, all receive TCO functionality is disabled, not just the packets that are directed to the BMC (also auto ARP packets).
RCV_ALL	1	Receive All Enable. 0b: Disable receiving all packets. 1b: Enable receiving all packets. Forwards all packets received over the wire that passed L2 filtering to the external BMC. This flag has no effect if bit 0 (Enable TCO packets) is disabled.
EN_STA	2	Enable Status Reporting. 0b: Disable status reporting. 1b: Enable status reporting.
Field	**Bit(s)**	**Description**
EN_ARP_RES	3	Enable ARP Response. 0b: Disable the 82598 ARP response. The 82598 treats ARP packets as any other packet, for example, packet is forwarded to the BMC if it passed other (non-ARP) filtering. 1b: Enable the 82598 ARP response. The 82598 automatically responds to all received ARP requests that match its IP address. Note that setting this bit does not change the Rx filtering settings. Appropriate Rx filtering to enable ARP request packets to reach the BMC should be set by the BMC or by the EEPROM. The BMC IP address is provided as part of the Receive Enable message (bytes 8-11). If a short version of the command is used, the 82598 uses IP address configured in the most recent long version of the command in which the EN_ARP_RES bit was set. If no such previous long command exists, then the 82598 uses the IP address configured in the EEPROM as ARP Response IPv4 Address in the pass-through LAN configuration structure. If the *CBDM* bit is set, the 82598 uses the BMC dedicated MAC address in ARP response packets. If the CBDM bit is not set, the BMC uses the host MAC address.
NM	5:4	Notification Method. Define the notification method the 82598 uses. 00b: SMBUS Alert. 01b: Asynchronous Notify. 10b: Direct Receive. 11b: Not Supported. **Note:** In dual SMBus address mode, both SMBus addresses **must** be configured with the same notification method.

Reserved	6	Reserved. Must be set to 1b.
CBDM	7	Configure the BMC Dedicated MAC Address. **Note:** This bit should be set to 0b when the *RCV_EN* bit (bit 0) is not set. 0b: The 82598 shares the MAC address for MNG traffic with the host MAC address specified in EEPROM words 2h:0h. 1b: The 82598 uses the BMC dedicated MAC address as a filter for incoming receive packets and as the sender address in ARP response packets. The BMC MAC address is set in bytes 7:2 in this command. If the short version of the command is used, the 82598 uses the MAC address configured in the most recent long version of the command in which the CBDM bit was set. If no such previous long command exists, then the 82572 uses the MAC address configured in the EEPROM as MAC address 3 (MMAL/H3) in the pass-through LAN configuration structure. When the Dedicated MAC address feature is activated, the 82598 uses the following registers to filter in all the traffic addressed to the BMC MAC. The BMC should not modify these registers as follows: Manageability Decision Filter - MDEF7 (and corresponding bit 7 in Management Control To Host Register - MANC2H), Manageability MAC Address Low - MMAL3 and Manageability MAC Address High - MMAH3 (and corresponding bit 3 of Manageability Filters Valid - MFVAL).

5.3.10.1.3.1 Management MAC Address (Data Bytes 7:2)

Ignored if the *CBDM* bit is not set. This MAC address is used to configure the dedicated MAC address. In addition, it is used in the ARP response packet when the EN_ARP_RES bit is set. This MAC address is also used when *CBDM* bit is set in subsequent short versions of this command.

5.3.10.1.3.2 Management IP Address (Data Bytes 11:8)

This IP address is used to filter ARP request packets.

5.3.10.1.3.3 Asynchronous Notification SMBus Address (Data Byte 12)

This address is used for the asynchronous notification SMBus transaction and for direct receive.

5.3.10.1.3.4 Interface Data (Data Byte 13)

Interface data byte used in asynchronous notification.

5.3.10.1.3.5 Alert Value Data (Data Byte 14)

Alert Value data byte used in asynchronous notification.

5.3.10.1.4 Force TCO Command

This command causes the 82598 to perform a TCO reset, if Force TCO reset is enabled in the EEPROM. The force TCO reset clears the data path (Rx/Tx) of the 82598 to enable the BMC to transmit/receive packets through the 82598. Note that in single -address mode, both ports are reset when the command is issued. In dual-address mode, force TCO reset is asserted only to the port related to the SMBus address the command was issued to. This command should only be used when the BMC is unable to transmit receive and suspects that the 82598 is inoperable. This command also causes the LAN device driver to unload. It is recommended to perform a system restart to resume normal operation.

The 82598 considers the Force TCO command as an indication that the operating system is hung and clears the *DRV_LOAD* flag. The Force TCO Reset command format is as follows:

Function	Command	Byte Count	Data 1
Force TCO Reset	0xCF	1	TCO Mode

Where TCO Mode is:

Field	Bit(s)	Description
DO_TCO_RST	0	Perform TCO Reset. 0b: Do nothing. 1b: Perform TCO reset.
Reserved	7:1	Reserved (set to 0x00).

5.3.10.1.5 Management Control

This command is used to set generic manageability parameters. The parameters list is shown in Management Control Command Parameters/Content. The command is 0xC1 which states that it is a Management Control command. The first data byte is the parameter number and the data after words (length and content) are parameter specific as shown in Management Control Command Parameters/ Content.

Note: If the parameter that the BMC sets is not supported by the 82598. The 82598 does not NACK the transaction. After the transaction ends, the 82598 discards the data and asserts a transaction abort status.

The Management Control command format is as follows:

Function	Command	Byte Count	Data 1	Data 2	...	Data N
Management Control	0xC1	N	Parameter Number	Parameter Dependent		

Table 5-5. Management Control Command Parameters/Content

Parameter	#	Parameter Data
Keep PHY Link Up	0x00	A single byte parameter: Data 2: Bit 0: Set to indicate that the PHY link for this port should be kept up throughout system resets. This is useful when the server is reset and the BMC needs to keep connectivity for a manageability session. Bit [7:1] Reserved. 0b: Disabled. 1b: Enabled.

5.3.10.1.6 Update Management Receive Filter Parameters

This command is used to set the manageability receive filters parameters. The command is 0xCC. The first data byte is the parameter number and the data that follows (length and content) are parameter specific as listed in management RCV filter parameters.

Note: If the parameter that the BMC sets is not supported by the 82598, then the 82598 does not NACK the transaction. After the transaction ends, the 82598 discards the data and asserts a transaction abort status.

The update management RCV receive filter parameters command format is as follows:

Function	Command	Byte Count	Data 1	Data 2	...	Data N
Update Manageability Filter Parameters	0xCC	N	Parameter Number	Parameter Dependent		

Management RCV filter parameters lists the different parameters and their content.

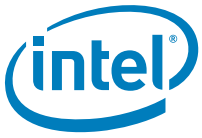


Table 5-6. Management RCV Filter Parameters

Parameter	Number	Parameter Data
Filters Enables	0x1	Defines the generic filters configuration. The structure of this parameter is four bytes as the MANC register. **Note:** The general filter enable is in the Receive Enable command that enables receive filtering.
Management-to-Host Configuration	0xA	This parameter defines which of the packet types identified as manageability packets in the receive path are directed to the host memory. Data 5:2 = MNG2H register bits.
Fail-Over Configuration	0xB	Fail-over register configuration. The bytes of this parameter are loaded into the Fail-Over Configuration register. See Section 5.3.11.3 for more information. Data 2:5 = Fail-Over Configuration register (bit 0 of Data 2 is bit 0 of the configuration register.).
Flex Filter 0 Enable Mask and Length	0x10	Flex Filter 0 Mask. Data 17:2 = Mask. Bit 0 in data 2 is the first bit of the mask. Data 19:18 = Reserved. Should be set to 00b. Date 20 = Flexible filter length.
Flex Filter 0 Data	0x11	Data 2 = Group of flex filter's bytes: 0x0 = bytes 0-29 0x1 = bytes 30-59 0x2 = bytes 60-89 0x3 = bytes 90-119 0x4 = bytes 120-127 Data 3:32 = Flex filter data bytes. Data 3 is LSB. Group's length is not a mandatory 30 bytes; it might vary according to filter's length and must NOT be padded by zeros. **Note:** Using this command to configure the filters data must be done after the flex filter mask command is issued and the mask is set.
Flex Filter 1 Enable Mask and Length	0x20	Same as parameter 0x10 but for filter 1.
Flex Filter 1 Data	0x21	Same as parameter 0x11 but for filter 1.
Flex Filter 2 Enable Mask and Length	0x30	Same as parameter 0x10 but for filter 2.
Flex Filter 2 Data	0x31	Same as parameter 0x11 but for filter 2.
Flex Filter 3 Enable Mask and Length	0x40	Same as parameter 0x10 but for filter 3.
Flex Filter 3 Data	0x41	Same as parameter 0x11 but for filter 3.
Parameter	**Number**	**Parameter Data**

Filters Valid	0x60	Four bytes to determine which of the 82598 filter registers contain valid data. Loaded into the MFVAL0 and MFVAL1 registers. Should be updated after the contents of a filter register are updated. Data 2: MSB of MFVAL. ... Data 5: LSB of MFVAL.
Decision Filters	0x61	Five bytes are required to load the manageability decision filters (MDEF). Data 2: Decision filter number. Data 3: MSB of MDEF register for this decision filter. ... Data 6: LSB of MDEF register for this decision filter.
VLAN Filters	0x62	Three bytes are required to load the VLAN tag filters. Data 2: VLAN filter number. Data 3: MSB of VLAN filter. Data 4: LSB of VLAN filter.
Flex Port Filters	0x63	Three bytes are required to load the manageability flex port filters. Data 2: Flex port filter number. Data 3: MSB of flex port filter. Data 4: LSB of flex port filter.
IPv4 Filters	0x64	Five bytes are required to load the IPv4 address filter. Data 2: IPv4 address filter number (3:0). Data 3: MSB of IPv4 address filter. … Data 6: LSB of IPv4 address filter.
IPv6 Filters	0x65	17 bytes are required to load the IPv6 address filter. Data 2: IPv6 address filter number (3:0). Data 3: MSB of IPv6 address filter. … Data 18: LSB of IPv6 address filter.
MAC Filters	0x66	Seven bytes are required to load the MAC address filters. Data 2: MAC address filters pair number (3:0). Data 3: MSB of MAC address. … Data 8: LSB of MAC address.

5.3.10.2 Read Transactions (82598 to BMC)

This section details the pass-through read transactions that the BMC can send to the 82598 over the SMBus.

SMBus read transactions lists the different SMBus read transactions supported by the 82598. All the read transactions are compatible with SMBus read block protocol format.

Table 5-7. SMBus Read Transactions

TCO Command	Transaction	Command	Opcode	Fragments	Section
Receive TCO Packet	Block Read	0xD0 or 0xC0	First: 0x90 Middle: 0x10 Last[1]: 0x50	Multiple	5.3.10.2.1
Read Status	Block Read	0xD0 or 0xC0 or 0xDE	Single: 0xDD	Single	5.3.10.2.2
Get System MAC Address	Block Read	0xD4	Single: 0xD4	Single	5.3.10.2.3
Read Configuration	Block Read	0xD2	Single: 0xD2	Single	5.3.10.2.4
Read Management Parameters	Block Read	0xD1	Single: 0xD1	Single	5.3.10.2.5
Read Management RCV Filter Parameters	Block Read	0xCD	Single: 0xCD	Single	5.3.10.2.6
Read Receive Enable Configuration	Block Read	0xDA	Single: 0xDA	Single	5.3.10.2.7

1. The last fragment of the receive TCO packet is the packet status.

0xC0 or 0xD0 commands are used for more than one payload. If BMC issues these read commands, and the 82598 has no pending data to transfer, it always returns as default opcode 0xDD with the 82598 status and does not NACK the transaction.

5.3.10.2.1 Receive TCO LAN Packet Transaction

The BMC uses this command to read packets received on the LAN and its status. When the 82598 has a packet to deliver to the BMC, it asserts the SMBus notification for the BMC to read the data (or direct receive). Upon receiving notification of the arrival of a LAN receive packet, the BMC begins issuing a Receive TCO packet command using the block read protocol.

A packet can be transmitted to the BMC in at least two fragments (at least one for the packet data and one for the packet status). As a result, BMC should follow the *F* and *L* bit of the op-code.

The op-code can have these values:

- 0x90 - First Fragment
- 0x10 - Middle Fragment
- When the opcode is 0x50, this indicates the last fragment of the packet, which contains packet status.

If a notification timeout is defined (in the EEPROM) and the BMC does not finish reading the whole packet within the timeout period, since the packet has arrived, the packet is silently discarded.

Following is the receive TCO packet format and the data format returned from the 82598.

Function	Command
Receive TCO Packet	0xC0 or 0xD0

Function	Byte Count	Data 1 (Op-Code)	Data 2	...	Data N
Receive TCO First Fragment	N	0x90	Packet Data Byte	...	Packet Data Byte
Receive TCO Middle Fragment	N	0x10	Packet Data Byte		
Receive TCO Last Fragment		0x50	Packet Data Byte		

5.3.10.2.1.1 Receive TCO LAN Status Payload Transaction

This transaction is the last transaction that the 82598 issues when a packet received from the LAN is transferred to the BMC. The transaction contains the status of the received packet.

The format of the status transaction is as follows:

Function	Byte Count	Data 1 (Op-Code)	Data 2 – Data 17 (Status Data)
Receive TCO Long Status	17 (0x11)	0x50	See Below

The status is 16 bytes where byte 0 (bits 7:0) is set in Data 2 of the status and byte 15 in Data 17 of the status.

TCO LAN packet status data lists the content of the status data.

Table 5-8. TCO LAN Packet Status Data

Name	Bits	Description
Packet Length	13:0	Packet length including CRC, only 14 LSB bits.
Reserved	24:14	Reserved.
CRC	25	CRC Insert (CRC insertion is needed).
Reserved	28:26	Reserved.
VEXT	29	Additional VLAN present in packet.
VP	30	VLAN Stripped (VLAN TAG insertion is needed).
Reserved	33:31	Reserved.
Flow	34	TX/RX Packet (Packet Direction (0b = Rx, 1b = Tx).
LAN	35	LAN number.
Reserved	39:36	Reserved.
Reserved	47:40	Reserved.
VLAN	63:48	The two bytes of the VLAN header tag.
Error	71:64	See Error Status Information.
Status	79:72	See Status Info.
Reserved	87:80	Reserved.
MNG Status	127:88	This field should be ignored if Receive TCO is not enabled (see Management Status).

Bit descriptions of each field in can be found in Section 4..

Table 5-9. Error Status Information

Field	Bits	Description
RXE	7	RX Data Error
IPE	6	IPv4 Checksum Error
TCPE	5	TCP/UDP Checksum Error
CXE	4	Carrier Extension Error
Rsv	3	Reserved
SEQ	2	Sequence Error
SE	1	Symbol Error
CE	0	CRC Error or Alignment Error

Table 5-10. Status Info

Field	Bits	Description
UDPV	7	Checksum field is valid and contains checksum of UDP fragment header
IPIDV	6	IP Identification Valid
CRC32V	5	CRC 32 valid bit indicates that the CRC32 check was done and a valid result was found
Reserved	4	Reserved
IPCS	3	IPv4 Checksum Calculated on Packet
TCPCS	2	TCP Checksum Calculated on Packet
UDPCS	1	UDP Checksum Calculated on Packet
Reserved	0	Reserved

Table 5-11. Management Status

Name	Bits	Description
Pass RMCP 0x026F	0	Set when the UDP/TCP port of the manageability packet is 0x26F.
Pass RMCP 0x0298	1	Set when the UDP/TCP port of the manageability packet is 0x298.
Pass MNG Broadcast	2	Set when the manageability packet is a broadcast packet.
Pass MNG Neighbor	3	Set when the manageability packet neighbor discovery packet.
Pass ARP Request/ARP Response	4	Set when the manageability packet is ARP response/request packet.
Reserved	7:5	Reserved.
Pass MNG VLAN Filter Index	10:8	
MNG VLAN Address Match	11	Set when the manageability packet match one of the MNG VLAN filters.
Unicast Address Index	14:12	Match any of the four unicast MAC address.
Unicast Address Match	15	Match any of the four unicast MAC address.
L4 port Filter Index	22:16	Indicate the flex filter number.
L4 port Match	23	Match any of the UDP/TCP port filters.
Flex TCO Filter Index	26:24	If bit 27 is set, this field indicates which TCO filter was matched.
Flex TCO Filter Match	27	Set if a flexible filter matched.
IP Address Index	29:28	IP filter number. (IPv4 or IPv6).
IP Address Match	30	Match any of the IP address filters.
IPv4/IPv6 Match	31	IPv4 match or IPv6 match. This bit is valid only if the bit 30 (IP match bit) or bit 4 (ARP match bit) are set.
Decision Filter Match	39:32	Match decision filter.

5.3.10.2.2 Read Status Command

The BMC should use this command after receiving a notification from the 82598 (such as SMBus Alert). The 82598 also sends a notification to the BMC in either of the following two cases:

- The BMC asserts a request for reading the status.
- The 82598 detects a change in one of the Status Data 1 bits (and was set to send status to the BMC on status change) in the Receive Enable command.

Note: Commands 0xC0/0xD0 are for backward compatibility and can be used for other payloads. The 82598 defines these commands in the opcode as well as which payload this transaction is. When the 0XDE command is set, the 82598 always returns opcode 0XDD with the 82598 status. The BMC reads the event causing the notification, using the Read Status command as follows:

Note: The 82598 response to one of the commands (0xC0 or 0xD0) in a given time as defined in the SMBus Notification Timeout and Flags word in the EEPROM.

Function	Command
Read Status	0XC0 or 0XD0 or 0XDE

Function	Byte Count	Data 1 (Op-Code)	Data 2 (Status Data 1)	Data 3 (Status Data 2)
Receive TCO Partial Status	3	0XDD	See Below	

Status Data Byte 1 lists the status data byte 1 parameters.

Table 5-12. Status Data Byte 1

Bit	Name	Description
7	Reserved	Reserved.
6	TCO Command Aborted	1b = A TCO command abort event occurred since the last read status cycle. 0b = A TCO command abort event did not occur since the last read status cycle.
5	Link Status Indication	0b = LAN link down. 1b = LAN link up[1].
4	PHY Link Forced Up	Contains the value of the *PHY_Link_Up* bit. When set, indicates that the PHY link is configured to keep the link up.
3	Initialization Indication	0b = An EEPROM reload event has not occurred since the last Read Status cycle. 1b = An EEPROM reload event has occurred since the last Read Status cycle[2].
2	Reserved	Reserved.
1:0	Power State	00b = Dr state. 01b = D0u state. 10b = D0 state. 11b = D3 state[3].

1. When the 82598 is operating in teaming mode, and presented as one SMBus device, the link indication is 0b only when both links (on both ports) are down. If one of the LANs is disabled, its link is considered to be down.
2. This indication is asserted when the 82598 manageability block reloads the EEPROM and its internal database is updated to the EEPROM default values. This is an indication that the external BMC should reconfigure the 82598, if other values other than the EEPROM default should be configured.
3. In single-address mode, the 82598 reports the highest power-state modes in both devices. The "D" state is marked in this order: D0, D0u, Dr, and D3.

Status data byte 2 is used by the BMC to indicate whether the LAN device driver is alive and running.

The LAN device driver valid indication is a bit set by the LAN device driver during initialization; the bit is cleared when the LAN device driver enters a Dx state or is cleared by the hardware on a PCI reset.

Bits 2 and 1 indicate that the LAN device driver is stuck. Bit 2 indicates whether the interrupt line of the LAN function is asserted. Bit 1 indicates whether the LAN device driver dealt with the interrupt line before the last Read Status cycle. Table 5-13 lists status data byte 2.

Table 5-13. Status Data Byte 2

Bit	Name	Description
5	Reserved	Reserved.
4	Reserved	Reserved.
3	Driver Valid Indication	0b = LAN driver is not alive. 1b = LAN driver is alive.
2	Interrupt Pending Indication	1b = LAN interrupt line is asserted. 0b = LAN interrupt line is not asserted.
1	ICR Register Read/Write	1b = ICR register was read since the last read status cycle. 0b = ICR register was not read since the last read status cycle. Reading the ICR indicates that the driver has dealt with the interrupt that was asserted.
0	Reserved	Reserved

Note: When the 82598 is in teaming mode, the bits listed in Status Data Byte 2 represent both cores:

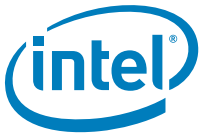
1. The LAN device driver alive indication is set if one of the LAN device drivers is alive.
2. The LAN interrupt is considered asserted if one of the interrupt lines is asserted.
3. The ICR is considered read if one of the ICRs was read (LAN 0 or LAN 1).

Status Data Byte 2 (bits 2 and 1) lists the possible values of bits 2 and 1 and what the BMC can assume from the bits:

Table 5-14. Status Data Byte 2 (Bits 2 and 1)

Previous	Current	Description
Don't Care	00b	Interrupt is not pending (OK).
00b	01b	New interrupt is asserted (OK).
10b	01b	New interrupt is asserted (OK).
11b	01b	Interrupt is waiting for reading (OK).
01b	01b	Interrupt is waiting for reading by the driver for more than one read cycle (not OK). Possible drive hang state.
Don't Care	11b	Previous interrupt was read and current interrupt is pending (OK).
Don't Care	10b	Interrupt is not pending (OK).

Note: The BMC reads should consider the time it takes for the LAN device driver to deal with the interrupt (in µs). Note that excessive reads by the BMC can give false indications.



5.3.10.2.3 Get System MAC Address

The Get System MAC Address returns the system MAC address over to the SMBus. This command is a single-fragment Read Block transaction that returns the following data:

Note: This command returns the MAC address configured in EEEPROM offset 0.

The format is as follows:

Function	Command
Get System MAC Address	0xD4

Data returned from the 82598:

Function	Byte Count	Data 1 (Op-Code)	Data 2	...	Data 7
Get System MAC Address	7	0xD4	MAC Address MSB	...	MAC Address LSB

5.3.10.2.4 Read Configuration

This command can be used by the BMC to read the CSR's from the manageability firmware.

In order to read a manageability CSR, the BMC executes two SMBus transactions. The first transaction is a block write that sets the CSR that the BMC needs to read. The second transaction is a block read that reads the CSR value.

The block write transaction is as follows:

Function	Command	Byte Count	Data 1	Data 2	Data 3
Write Configuration	0xC6	3	CSR Number MSB	...	CSR Number LSB

Following the block write, the BMC issues a block read that reads the parameter that was set in the Block Write command:

Function	Command
Read Configuration	0xD2

For example, if the BMC wants to read the MANC register, it would issue a block write with command of 0xC6, length of 3 and parameters of 0x00, 0x58, 0x20.

5.3.10.2.5 Read Management Parameters

In order to read the management parameters, the BMC executes two SMBus transactions. The first transaction is a block write that sets the parameter that the BMC needs to read. The second transaction is a block read that reads the parameter.

The block write transaction is as follows:

Function	Command	Byte Count	Data 1
Management Control Request	0xC1	1	Parameter Number

Following the block write, the BMC issues a block read that reads the parameter that was set in the Block Write command:

Function	Command
Read Management Parameter	0xD1

Data returned from the 82598:

Function	Byte Count	Data 1 (Op-Code)	Data 2	Data 3	...	Data N
Read Management Parameter	N	0xD1	Parameter Number	Parameter Dependent		

The returned data is in the same format of the Management Control command.

Note that it might be that the parameter that is returned is not the parameter requested by the BMC. The BMC should verify the parameter number (default parameter to be returned is 0x10).

If the parameter number is 0xFF, it means that the data that the 82598 should supply is not ready yet. The BMC should retry the read transaction.

It is the BMC responsibility to follow the previous procedure. In the case where the BMC sends a Block Read command, which is not preceded by a Block Write command with byte count = 1b, the 82598 sets the parameter number in the read block transaction to be 0xFE.

5.3.10.2.6 Read Management Receive Filter Parameters

In order to read the Management RCV filter parameters, the BMC should execute two SMBus transactions. The first transaction is a block write that sets the parameter that the BMC wants to read. The second transaction is block read that read the parameter.

Following is the block write transaction:

Function	Command	Byte Count	Data 1	Data 2
Update MNG RCV Filter Parameters	0xCC	1 or 2	Parameter Number	Parameter Data

Following the block write, the BMC should issue a block read that reads the parameter that was set in the Block Write command as follows:

Parameter	#	Parameter Data
Filters Enable	0x01	None
MNG2H Configuration	0x0A	None
Fail-Over Configuration	0x0B	None
Flex Filter 0 Enable Mask and Length	0x10	None
Flex Filter 0 Data	0x11	Data 2: Group of Flex Filter's Bytes: 0x0 = bytes 0-29 0x1 = bytes 30-59 0x2 = bytes 60-89 0x3 = bytes 90-119 0x4 = bytes 120-127
Flex Filter 1 Enable Mask and Length	0x20	None
Flex Filter 1 Data	0x21	Same as parameter 0x11 but for filter 1.
Flex Filter 2 Enable Mask and Length	0x30	None
Flex Filter 2 Data	0x31	Same as parameter 0x11 but for filter 2.
Parameter	**#**	**Parameter Data**
Flex Filter 3 Enable Mask and Length	0x40	None
Flex Filter 3 Data	0x41	Same as parameter 0x11 but for filter 3.
Filters Valid	0x60	None
Decision Filters	0x61	One byte to define the accessed manageability decision filter (MDEF) Data 2 – Decision Filter number
VLAN Filters	0x62	One byte to define the accessed VLAN tag filter (MAVTV) Data 2 – VLAN Filter number
Flex Ports Filters	0x63	One byte to define the accessed manageability flex port filter (MFUTP). Data 2 – Flex Port Filter number

IPv4 Filter	0x64	One byte to define the accessed IPv4 address filter (MIPAF) Data 2 – IPv4 address filter number
IPv6 Filters	0x65	One byte to define the accessed IPv6 address filter (MIPAF) Data 2 – IPv6 address filter number
MAC Filters	0x66	One byte to define the accessed MAC address filters pair (MMAL, MMAH) Data 2 – MAC address filters pair number (0-3)

Following the block write, the BMC issues a block read that reads the parameter that was set in the Block Write command:

Function	Command
Request MNG RCV Filter Parameters	0xCD

Data returned from the 82598:

Function	Byte Count	Data 1 (Op-Code)	Data 2	Data 3	...	Data N
Read MNG RCV Filter Parameters	N	0xCD	Parameter Number	Parameter Dependent		

The returned data is the same format as the Update command.

Note that it might be that the parameter that is returned is not the parameter requested by the BMC. The BMC should verify the parameter number (default parameter to be returned is 0x1).

Note: If the parameter number is 0xFF then this indicates that the data the 82598 should supply is not ready yet and the BMC should retry the read transaction.

It is the BMC responsibility to follow the procedure previously defined. In the case where the BMC sends a Block Read command that is not preceded by a Block Write command with byte count = 1b, the 82598 sets the parameter number in the read block transaction as 0xFE.

5.3.10.2.7 Read Receive Enable Configuration

The BMC uses this command to read the receive configuration data. This data can be configured when using Receive Enable command or through the EEPROM.

Read Receive Enable Configuration command format (SMBus Read Block) is as follows:

Function	Command
Read Receive Enable	0xDA

Data returned from the 82598:

Function	Byte Count	Data 1 (Op-Code)	Data 2	Data 3	...	Data 8	Data 9	...	Data 12	Data 13	Data 14	Data 15
Read Receive Enable	15 (0x0F)	0xDA	Receive Control Byte	MAC Addr LSB	...	MAC Addr MSB	IP Addr LSB	...	IP Addr MSB	BMC SMBus Addr	I/F Data Byte	Alert Value Byte

5.3.11 LAN Fail-Over in LAN Teaming Mode

Manageability fail-over is the ability in a dual-port network device (the 82598) to detect that the LAN connection on the manageability enabled port is lost, and to enable the other port in the device to receive/transmit manageability packets. When the 82598 operates in teaming mode, the OS and the external BMC consider the 82598 as one logical network device. The decision to determine which of the 82598 ports to use is done internally in the 82598 (or in the ANS driver in case of the regular receive/transmit traffic). This section deals with fail-over in teaming mode only. In non-teaming mode, the external BMC should consider the 82598's network ports as two different network devices, and is solely responsible for the fail-over mechanism.

5.3.11.1 Fail-Over Functionality

In teaming mode, the 82598 mirrors both the network ports into a single SMBus slave device. The 82598 automatically handles the configurations of both network ports. Thus, for configurations, receiving and transmitting the BMC should consider both ports as a single entity.

When the currently active port for transmission becomes unavailable (for instance, the link is down), the 82598 automatically tries to switch the packet transmission to the other port. Thus, as long as one of the ports is valid, the BMC has a valid link indication for the SMBus slave.

5.3.11.1.1 Transmit Functionality

In order to transmit a packet, the BMC should issue the appropriate SMBus packet transmission commands to the 82598. The 82598 will then automatically choose the transmission port.

5.3.11.1.2 Receive Functionality

When the 82598 receives a packet from any of the teamed ports, it will notifies and forwards the packet to the BMC.

Note: As both ports might be active (for instance, with a valid link), packets might be received on the currently non-active port. To avoid this, fail-over should be used only in a switched network.

5.3.11.1.3 Port Switching (Fail-Over)

While in teaming mode, transmit traffic is always transmitted by the 82598 through only one of the ports at any given time. The 82598 might switch the traffic transmission between ports under any of the following conditions:

1. The current transmitting port link is not available.
2. The preferred primary port is enabled and becomes available for transmission.

5.3.11.1.4 Device Driver Interactions

When the LAN device driver is present, the decision to switch between the two ports is done by the device driver. When the device driver is absent, this decision is done internally by the 82598.

Note: When the device driver releases teaming mode, such as when the system state changes, the 82598 reconfigures the LAN ports to teaming mode. The 82598 accomplishes this by re-setting the MAC address of the two ports to be the teaming address in order to re-start teaming. This is followed by transmitting gratuitous ARP packets to notify the network of teaming mode re-setting.

5.3.11.2 Fail-Over Configuration

Fail-over operation is configured through the fail-over register, as described in Table 5-15.

The BMC should configure this register after every initialization indication from the 82598 (such as after every the 82598 firmware reset). The BMC needs to use the Update Management Receive Filters command, with parameter 0x0A, detailed in Section 5.3.10.1.6.

The different configurations available to the BMC are detailed in this section.

Note: In teaming mode, both ports should be configured with the same receive manageability filters parameters (EEPROM sections for port 0 and port 1 should be identical).

5.3.11.2.1 Preferred Primary Port

The BMC might choose one of the network ports (LAN0 or LAN1) as a preferred primary port for packet transmission. The 82598 uses the preferred primary port as the transmission port each time the link for that port is valid. For example, the 82598 always switches back to the preferred primary port when available.

5.3.11.2.2 Gratuitous ARPs

In order to notify the link partner that a port switching has occurred, the 82598 can be configured to automatically send gratuitous ARPs. These gratuitous ARPs cause the link partner to update its ARP tables to reflect the change.

The BMC might enable/disable gratuitous ARPs, configure the number of gratuitous ARPs, or the interval between them by modifying the fail-over register.

5.3.11.2.3 Link Down Timeout

The BMC can control the timeout for a link to be considered invalid. The 82598 waits on this timeout before attempting to switch from an inactive port.

5.3.11.3 Fail-Over Register

This register is loaded at power up from the EEPROM. The BMC can change the contents of the fail-over register using the Update Management Receive Filters command, with parameter 0x0A, detailed in Section 5.3.10.1.6.

Table 5-15 lists the register different bits:

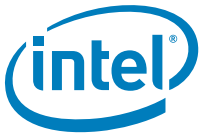


Table 5-15. Fail-Over Register

Bits	Field	Initial Value	Read/ Write	Description
0	RMP0EN	0x1	RO	RCV MNG port 0 Enable. When this bit is set, it reports that management traffic will be received from port 0.
1	RMP1EN	0x1	RO	RCV MNG port 1 Enable. When this bit is set, it reports that management traffic will be received from port 1.
2	MXP	0x0	RO	MNG XMT Port. 0b - reports that management traffic should be transmitted through port 0. 1b – reports that MNG traffic should be transmitted through port 1.
3	PRPP	0x0	RW	Preferred Primary Port. 0b – Port 0 is the preferred primary port. 1b – Port 1 is the preferred primary port.
4	PRPPE	0x0	RW	Preferred primary port enables.
5	Reserved	0x0	RO	Reserved
6	RGAEN	0x0		Repeated Gratuitous ARP Enable. If this bit is set, the 82598 sends a configurable number of gratuitous ARP packets (GAC bits of this register) using configurable interval (GATI bits of this register) after the following events: • System move to Dx. • Fail-over event initiated the 82598.
8:7	Reserved	0x0	RO	Reserved
Bits	**Field**	**Initial Value**	**Read/ Write**	**Description**
9	TFOENODX	0x0	RW	Teaming Fail-Over Enable on Dx. Enable fail-over mechanism. Bits 3:8 are valid only if this bit is set.
10:11	Reserved	0x0	RO	Reserved
12:15	GAC	0x0	RW	Gratuitous ARP Counter. Indicates the number of gratuitous ARP that should be sent after a fail-over event and after move to Dx. The value of 0b means that there is no limit on the gratuitous ARP packets to be sent.
16:23	LDFOT	0x0	RW	Link down Fail-Over Time. Defines the time (in seconds) the link should be down before doing a fail-over to the second port. This is also the time that the primary link should be up (after it was down) before the 82598 will fail-over back to the primary port.
24:31	GATI	0x0	RW	Gratuitous ARP Transmission Interval. Defines the interval in seconds before retransmission of gratuitous ARP packets.

5.3.12 SMBus Troubleshooting Guide

This section outlines the most common issues found while working with pass-through using the SMBus sideband interface.

5.3.12.1 TCO Alert Line Stays Asserted After a Power Cycle

After the 82598 resets both of its ports indicates a status change. If the BMC only reads status from one port (slave address) the other one will continue to assert the TCO alert line.

Ideally, the BMC should use the ARA transaction (see Section 5.3.9) to determine which slave asserted the TCO alert. Many customers only wish to use one port for manageability thus using ARA might not be optimal.

An alternate to using ARA is to configure one of the ports to not report status and to set its SMBus timeout period. In this case, the SMBus timeout period determines how long a port asserts the TCO alert line awaiting a status read from a BMC; by default this value is zero, which indicates an infinite timeout.

The SMBus configuration section of the EEPROM has a SMBus Notification Timeout (ms) field that can be set to a recommended value of 0xFF (for this issue). Note that this timeout value is for both slave addresses. Along with setting the SMBus Notification Timeout to 0xFF, it is recommended that the second port be configured in the EEPROM to disable status alerting. This is accomplished by having the *Enable Status Reporting* bit set to 0b for the desired port in the LAN configuration section of the EEPROM.

The last solution for this issue is to have the BMC hard-code the slave addresses to always read from both ports. As with the previous solution, it is also recommend that the second port have status reporting disabled.

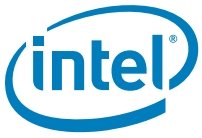
5.3.12.2 SMBus Commands are Always NACK'd by the 82598

There are several reasons why all commands sent to the 82598 from a BMC could be NACK'd. The following are the most common:

- Invalid EEPROM Image - The image itself might be invalid, or it could be a valid image; however, it is not a pass-through image, as such SMBus connectivity is disabled.
- The BMC is not using the correct SMBus address - Many BMC vendors hard-code the SMBus address(es) into their firmware. If the incorrect values are hard-coded, the 82598 does not respond.
  - The SMBus address(es) can also be dynamically set using the SMBus ARP mechanism.
- The BMC is using the incorrect SMBus interface - The EEPROM might be configured to use one physical SMBus port; however, the BMC is physically connected to a different one.
- Bus Interference - the bus connecting the BMC and the 82598 might be unstable.

5.3.12.3 SMBus Clock Speed is 16.6666 KHz

This can happen when the SMBus connecting the BMC and the 82598 is also tied into another device (such as a ICH6) that has a maximum clock speed of 16.6666 KHz. The solution is to not connect the SMBus between the 82598 and the BMC to this device.



5.3.12.4 A Network Based Host Application is not Receiving any Network Packets

Reports have been received about an application not receiving any network packets. The application in question was NFS under Linux. The problem was that the application was using the RMPC/RMCP+ IANA reserved port 0x26F (623), and the system was also configured for a shared MAC and IP address with the OS and BMC.

The management control to host configuration, in this situation, was setup not to send RMCP traffic to the OS (this is typically the correct configuration). This means that no traffic send to port 623 was being routed.

The solution in this case is to configure the problematic application NOT to use the reserved port 0x26F.

5.3.12.5 Status Registers

If the EEPROM image is configured correctly, the physical connections are valid, and problems still exist, use LANConf or other utilities/drivers to check the appropriate 82598 status registers for other indications.

5.3.12.5.1 Firmware Semaphore Register (FWSM, 0x10148)

This register provides a way to find out if the firmware on the 82598 is functioning properly and if so, in what mode.

Check the error indication bits (24:19), if they are anything other than zero, then the firmware is not going to be fully functional, if at all.

The most common errors are:

- EEPROM checksum errors - these can be caused by a number of things:
  - Mismatch in 82598 stepping and EEPROM image version (old EEPROM image on a new 82598)
  - EEPROM part too small (recommended minimum size for manageability is 32 Kb)
  - Old utility was used to update the EEPROM (always make sure to have the latest versions)
- Invalid Firmware Mode (0x08)

If bits 3:1 of the register indicate a firmware mode that is reserved, this error condition can be reset.

Always make note of the firmware mode, bits 3:1. In nearly all cases, this value should be set to 010b for pass-through mode to an external BMC.

The firmware valid bit (15) should be set to 1b to indicate that the firmware is up and running. If it is not set to 1b, then an error code should be indicated in bits 24:19.

The reset count bits (18:16) indicate how many times the internal firmware on the 82598 has been reset. This value should be a one (the firmware was reset at power up). If the value is greater than one then there are issues somewhere. Note that this counter goes from 0-7 and wraps around.

5.3.12.5.2 Management Control Register (MANC 0x5820)

This register indicates which filters are enabled. It is possible to configure all of the filters yet not enable them, in which case, no management traffic is routed to the BMC. Or, the BMC might be receiving undesired traffic, such as ARP requests when the 82598 was configured to do automatic ARP responses.

Check this register if getting unwanted traffic or if packets aren't getting sent to the BMC.

Bit 17 (*Receive TCO Packets Enable*) must also be set in order for any packets are sent to a BMC. Note that it doesn't matter what the other enabled filters are, if this one is off, no packets are sent to the BMC.

Bit 21 (*Enable Management-to-Host*) enables or disables the various filters that also allow manageability traffic (all those that pass the filters in the 82598) to optionally be passed to the OS.

5.3.12.5.3 Management Control To Host Register (MANC2H 0x5860)

The 82598 has a large number of filtering mechanisms by which network traffic can be directed to a BMC. Traffic sent to the BMC is typically not sent to the host OS as well. For example, the OS is not interested in RMCP/RMCP+ traffic. However, the OS might be interested in ARP requests. If the 82598 is configured for automatic ARP requests, this means that a filter for ARP requests has been configured and enabled, if the ARP request matches that of the BMC, then that ARP request is not sent to the OS unless specifically configured to do so. This is what the MANC2H register shows, which manageability filters to also pass the data up to the OS as well as the BMC.

Using the ARP request example; typically it is desirable to allow the OS to receive these, as such, bit 7 (*ARP Request*) should be set.

5.3.12.6 Unable to Transmit Packets from the BMC

If the BMC has been transmitting and receiving data without issue for a period of time and then begins to receive NACKs from the 82598 when it attempts to write a packet, the problem is most likely due to the fact that the buffers internal to the 82598 are full of data that has been received from the network; however, has yet to be read by the BMC.

Being an embedded device, the 82598 has limited buffers, which it shares for receiving and transmitting data. If a BMC does not keep the incoming data read, the 82598 can be filled up, which does not allow the BMC to transmit anymore data, resulting in NACKs.

If this situation occurs, the recommended solution is to have the BMC issue a Receive Enable command to disable anymore incoming data, go read all the data from the 82598 and then use the Receive Enable command to enable incoming data once again.

5.3.12.7 SMBus Fragment Size

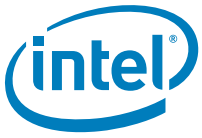
The SMBus specification indicates a maximum SMBus transaction size of 32 bytes. Most of the data passed between the 82598 and the BMC over the SMBus is RMCP/RMCP+ traffic, which by its very nature (UDP traffic) is significantly larger than 32 bytes in length, thus requiring multiple SMBus transactions to move a packet from the 82598 to the BMC or to send a packet from the BMC to the 82598.

Recognizing this bottleneck, the 82598 can handle up to 240 bytes of data within a single transaction. This is a configurable setting within the EEPROM.

The default value in the EEPROM images is 32, per the SMBus specification. If performance is an issue, it is recommended that you increase this size.

During the initialization phase, the firmware within the 82598 allocates buffers based upon the SMBus fragment size setting within the EEPROM. The 82598 firmware has a finite amount of RAM for its use, as such the larger the SMBus fragment size, the fewer buffers it can allocate. As such, the BMC implementation must take care to send data over the SMBus in an efficient way.

For example, the 82598 firmware has 3 KB of RAM it can use for buffering SMBus fragments. If the SMBus fragment size is 32 bytes then the firmware could allocate 96 buffers of size 32 bytes each. As a result, the BMC could then send a large packet of data (such as KVM) that is 800 bytes in size in 25 fragments of size 32 bytes apiece.



However, this might not be the most efficient way because the BMC must break the 800 bytes of data into 25 fragments and send each one at a time.

If the SMBus fragment size is changed to 240 bytes, the 82598 firmware can create 12 buffers of 240 bytes each to receive SMBus fragments. The BMC can now send that same 800 bytes of KVM data in only four fragments, which is much more efficient.

The problem of changing the SMBus fragment size in the EEPROM is if the BMC does not also reflect this change. If a programmer changes the SMBus fragment size in the 82598 to 240 bytes and then wants to send 800 bytes of KVM data, the BMC can still only send the data in 32 byte fragments. As a result, the firmware runs out of memory.

This is because the 82598 firmware created the 12 buffers of 240 bytes each for fragments, however the BMC is only sending fragments of size 32 bytes. This results in a memory waste of 208 bytes per fragment in this case, and when the BMC attempts to send more than 12 fragments in a single transaction, the 82598 NACKs the SMBus transaction due to not enough memory to store the KVM data.

In summary, if a programmer increases the size of the SMBus fragment size in the EEPROM, which is recommended for efficiency purposes, take care to ensure that the BMC implementation reflects this change and uses that fragment size to its fullest when sending SMBus fragments.

5.3.12.8 Enable XSum Filtering

If XSum filtering is enabled, the BMC does not need to perform the task of checking this checksum for incoming packets. Only packets that have a valid XSum is passed to the BMC, all others are silently discarded.

This is a way to offload some work from the BMC.

5.3.12.9 Still Having Problems?

If problems still exist, contact your field representative. Before contacting, be prepared to provide the following:

- The contents of status registers:
  - 0x5820
  - 0x5860
  - 0x10148
- A SMBus trace if possible
- A dump of the EEPROM image
  - This should be taken from the actual 82598, rather than the EEPROM image provided by Intel. Parts of the EEPROM image are changed after writing, such as the physical EEPROM size. This information could be key in helping assist in solving an issue.

5.3.13 Sample Configurations

This section provides an overview and sample settings for commonly used filtering configurations. Three examples are presented. The examples are in pseudo code format, with the name of the SMBus command, followed by the parameters for that command and an explanation.

Here is a sample:

Receive Enable[00]

Utilizing the simple form of the Receive Enable command, this prevents any packets from reaching the BMC by disabling filtering.

Example 5-1. Shared MAC, RMCP-Only Ports

This example will be the most basic configuration. The MAC address filtering will be shared with the host operating system and only traffic directed the RMCP ports (26Fh & 298h) will be filtered. For this simple example, the BMC must issue gratuitous ARPs because no filter will be enabled to pass ARP requests to the BMC.

Pseudo Code

Step 1: Disable existing filtering

Receive Enable[00]

Utilizing the simple form of the Receive Enable command, this prevents any packets from reaching the MC by disabling filtering:

Receive Enable Control 40h:

Bit 0 [0]– Disable Receiving of packets

Bit 6 [1] - Reserved, must be set to 1

Step 2: Configure MDEF[0]

```
Update Manageability Filter Parameters [61, 0, 00000C00]
```

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 61h). This will update MDEF[0], as indicated by the 2nd parameter (0).

MDEF[0] value of 00000C00h:

Bit 10 [1]– port 298h

Bit 11 [1]– port 26Fh

Step 3: Enable Filtering

```
Receive Enable [45]
```

Using the simple form of the Receive Enable command:

Receive Enable Control 05h:

Bit 0 [1] – Enable Receiving of packets

Bit 2 [1] – Enable status reporting (such as link lost)

Bit 5:4 [00]– Notification method = SMB Alert

Bit 6 [1] - Reserved, must be set to 1

Bit 7 [0]– Use shared MAC

Table 5-16. MDEF Results

Filter		Manageability Decision Filter (MDEF)							
		0	1	2	3	4	5	6	7
L2 Unicast Address	AND								
Broadcast	AND								
Manageability VLAN	AND								
IP Address	AND								
L2 Unicast Address	OR								
Broadcast	OR								
Multicast	AND								
ARP Request	OR								
ARP Response	OR								
Neighbor Discovery	OR								
Port 0x298	OR	x							
Port 0x26F	OR	x							
Flex Port 15:0	OR								
Flex TCO 3:0	OR								

Example 5-2. Dedicated MAC, Auto ARP Response and RMCPport Filtering

This example shows a common configuration; the BMC has a dedicated MAC and IP address. Automatic ARP responses will be enabled as well as RMCP port filtering. By enabling Automatic ARP responses the BMC is not required to send the gratuitous ARPs as it did in the previous example. Since ARP requests are now filtered, in order for the host to receive the ARP requests, the Manageability to Host filter will be configured to send the ARP requests to the host as well.

For demonstration purposes, the dedicated MAC address will be calculated by reading the System MAC address and adding 1 do it, assume the System MAC is AABBCCDC. The IP address for this example will be 1.2.3.4.

Additionally, the XSUM filtering will be enabled.

Note that not all Intel Ethernet Controllers support automatic ARP responses, please refer to product specific documentation.

Pseudo Code

Step 1: Disable existing filtering

```
Receive Enable[40]
```

Utilizing the simple form of the Receive Enable command, this prevents any packets from reaching the MC by disabling filtering:

Receive Enable Control 40h:

Bit 0 [0]– Disable Receiving of packets

Bit 6 [1] - Reserved, must be set to 1

Step 2: Read System MAC Address

```
Get System MAC Address []
```

Reads the System MAC address. Assume returned AABBCCDDED for this example.

Step 3: Enable MNG2Host & XSUM Filters

```
Update Manageability Filter Parameters [01, 00A00000]
```

Use the Update Manageability Filter Parameters command to update Filters Enable settings (parameter 1). This set the Manageability Control (MANC) Register.

MANC Register 00A00000h:

Bit 21 [1]- MNG2Host Filter Enable

Bit 23 [1]– XSUM Filter enable

Some of the following configuration steps manipulate the MANC register indirectly, this command sets all bits except XSUM to 0. It is important to either do this step before the others, or to read the value of the MANC and then write it back with only bit 32 changed. Also note that the XSUM enable bit may differ between Ethernet Controllers, refer to product specific documentation.

Step 4: Configure MDEF[0]

```
Update Manageability Filter Parameters [61, 0, 00000C00]
```

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 61h). This will update MDEF[0], as indicated by the 2nd parameter (0).

MDEF value of 00000C00h:

Bit 10 [1]– port 298h

Bit 11 [1]– port 26Fh

Step 5: Configure MDEF[1]

```
Update Manageability Filter Parameters [61, 1, 00000080]
```

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 61h). This will update MDEF[1], as indicated by the 2nd parameter (1).

MDEF value of 00000080:

Bit 7 [7]– ARP Requests

When Enabling Automatic ARP responses, the ARP requests still go into the manageability filtering system and as such need to be designated as also needing to be sent to the host. For this reason a separate MDEF is created with only ARP request filtering enabled.

Refer to the next step for details.

Step 6: Configure the Management to Host Filter

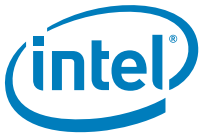
```
Update Manageability Filter Parameters [0A, 00000002]
```

Use the Update Manageability Filter Parameters command to update the Management Control to Host (MANC2H) Register.

MANC2H Register 00000002:

Bit 2 [1]– Enable MDEF[1] traffic to go to Host as well

This allows ARP requests to be passed to both manageability and to the host. Specified separate MDEF filter for ARP requests. If ARP requests had been added to MDEF[0] and then MDEF[0] specified in Management to Host configuration then not only would ARP requests be sent to the MC and host, RMCP



traffic (ports 26Fh and 298h) would have also been sent to both places.

The MANC2H Filter is configured in this step and enabled in step 3.

Step 7: Enable Filtering

```
Receive Enable [CD, AABBCCDDEE, 01020304, 00, 00, 00]
```

Using the advanced version Receive Enable command, the first parameter:

Receive Enable Control CDh:

Bit 0 [1] – Enable Receiving of packets

Bit 2 [1] – Enable status reporting (such as link lost)

Bit 3 [1] – Enable Automatic ARP Responses

Bit 5:4 [00]– Notification method = SMB Alert

Bit 6 [1] - Reserved, must be set to 1

Bit 7 [1]– Use Dedicated MAC

Second parameter is the MAC address (AABBCCDDEE).

Third Parameter is the IP address(01020304).

The last three parameters are zero when the notification method is SMB Alert.

Table 5-17. MDEF Results

Filter		Manageability Decision Filter (MDEF)							
		0	1	2	3	4	5	6	7
L2 Unicast Address	AND								x
Broadcast	AND								
Manageability VLAN	AND								
IP Address	AND								
L2 Unicast Address	OR								
Broadcast	OR								
Multicast	AND								
ARP Request	OR		x						
ARP Response	OR								
Neighbor Discovery	OR								
Port 0x298	OR	x							
Port 0x26F	OR	x							
Flex Port 15:0	OR								
Flex TCO 3:0	OR								

Example 5-3. Dedicated MAC & IP Address

This example provided the BMC with a dedicated MAC and IP address and allows it to receive ARP requests. The BMC is then responsible for responding to ARP requests.

For demonstration purposes, the dedicated MAC address will be calculated by reading the System MAC address and adding 1 do it, assume the System MAC is AABBCCDC. The IP address for this example will be 1.2.3.4. For this example, the Receive Enable command is used to configure the MAC address filter.

In order for the BMC to be able to receive ARP Requests, it will need to specify a filter for this, and that filter will need to be included in the Manageability To Host filtering so that the host OS may also receive ARP Requests.

Pseudo Code

Step 1: Disable existing filtering

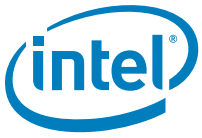
```
Receive Enable[40]
```

Utilizing the simple form of the Receive Enable command, this prevents any packets from reaching the MC by disabling filtering:

Receive Enable Control 40h:

Bit 0 [0]– Disable Receiving of packets

Bit 6 [1] - Reserved, must be set to 1



Step 2: - Read System MAC Address

```
Get System MAC Address []
```

Reads the System MAC address. Assume returned AABBCCDDED for this example.

Step 3: Configure IP Address Filter

```
Update Manageability Filter Parameters [64, 00, 01020304]
```

Use the Update Manageability Filter Parameters to configure an IPv4 filter.

The 1st parameter (64h) specifies that we are configuring an IPv4 filter.

The 2nd parameter (00h) indicates which IPv4 filter is being configured, in this case filter 0.

The 3rd parameter is the IP address – 1.2.3.4.

Step 4: Configure MAC Address Filter

```
Update Manageability Filter Parameters [66, 00, AABBCCDDEE]
```

Use the Update Manageability Filter Parameters to configure a MAC Address filter.

The 1st parameter (66h) specifies that we are configuring a MAC Address filter.

The 2nd parameter (00h) indicates which MAC Address filter is being configured, in this case filter 0.

The 3rd parameter is the MAC Address - AABBCCDDEE

Step 5: Configure MDEF[0] for IP and MAC Filtering

```
Update Manageability Filter Parameters [61, 0, 00000009]
```

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 61h). This will update MDEF[0], as indicated by the 2nd parameter (0).

MDEF value of 00000009:

Bit 1 [1]- MAC Address Filtering

Bit 3 [1]– IP Address Filtering

Step 6: Configure MDEF[1]

```
Update Manageability Filter Parameters [61, 1, 00000080]
```

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 61h). This will update MDEF[1], as indicated by the 2nd parameter (1).

MDEF value of 00000080:

Bit 7 [7]– ARP Requests

When filtering ARP requests the requests go into the manageability filtering system and as such need to be designated as also needing to be sent to the host. For this reason a separate MDEF is created with only ARP request filtering enabled.

Step 7: Configure the Management to Host Filter

```
Update Manageability Filter Parameters [0A, 00000002]
```

Use the Update Manageability Filter Parameters command to update the Management Control to Host (MANC2H) Register.

MANC2H Register 00000002:

Bit 2 [1]– Enable MDEF[1] traffic to go to Host as well

Step 8: Enable MNG2Host Filter

Update Manageability Filter Parameters [01, 00200000]

Use the Update Manageability Filter Parameters command to update Filters Enable settings (parameter 1). This set the Manageability Control (MANC) Register.

MANC Register 00200000h:

Bit 21 [1]- MNG2Host Filter Enable

This enables the MANC2H filter configured in step 7.

Step 9: Enable Filtering

Receive Enable [45]

Using the simple form of the Receive Enable command,:

Receive Enable Control 45h:

Bit 0 [1] – Enable Receiving of packets

Bit 2 [1] – Enable status reporting (such as link lost)

Bit 5:4 [00]– Notification method = SMB Alert

Bit 6 [1] - Reserved, must be set to 1

The Resulting MDEF filters are as follows:

Table 5-18. MDEF Results

		Manageability Decision Filter (MDEF)							
Filter		0	1	2	3	4	5	6	7
L2 Unicast Address	AND	x							
Broadcast	AND								
Manageability VLAN	AND								
IP Address	AND	x							
L2 Unicast Address	OR								
Broadcast	OR								
Multicast	AND								
ARP Request	OR		x						
ARP Response	OR								
Neighbor Discovery	OR								
Port 0x298	OR								
Port 0x26F	OR								
Flex Port 15:0	OR								
Flex TCO 3:0	OR								

Example 5-4. Dedicated MAC and VLAN Tag

This example shows an alternate configuration; the BMC has a dedicated MAC and IP address, along with a VLAN tag of 32h will be required for traffic to be sent to the BMC. This means that all traffic with VLAN a matching tag will be sent to the BMC.

For demonstration purposes, the dedicated MAC address will be calculated by reading the System MAC address and adding 1 do it, assume the System MAC is AABBCCDC. The IP address for this example will be 1.2.3.4 and the VLAN tag will be 0032h.

It is assumed the host will not be using the same VLAN tag as the BMC. If they were to share the same VLAN tag then additional filtering would need to be configured to allow VLAN tagged non-unicast (such as ARP requests) to be sent to the host as well as the BMC using the Manageability to Host filter capability.

Additionally, the XSUM filtering will be enabled.

Pseudo Code

Step 1: Disable existing filtering

```
Receive Enable[40]
```

Utilizing the simple form of the Receive Enable command, this prevents any packets from reaching the MC by disabling filtering:

Receive Enable Control 40h:

Bit 0 [0]– Disable Receiving of packets

Bit 6 [1] - Reserved, must be set to 1

Step 2: Read System MAC Address

```
Get System MAC Address []
```

Reads the System MAC address. Assume returned AABBCCDDED for this example.

Step 3: Configure XSUM Filter

```
Update Manageability Filter Parameters [01, 00800000]
```

Use the Update Manageability Filter Parameters command to update Filters Enable settings (parameter 1). This set the Manageability Control (MANC) Register.

MANC Register 00800000h:

Bit 23 [1]– XSUM Filter enable

Note that some of the following configuration steps manipulate the MANC register indirectly, this command sets all bits except XSUM to 0. It is important to either do this step before the others, or to read the value of the MANC and then write it back with only bit 32 changed. Also note that the XSUM enable bit may differ between Ethernet Controllers, refer to product specific documentation.

Step 4: Configure VLAN 0 Filter

```
Update Manageability Filter Parameters [62, 0, 0032]
```

Use the Update Manageability Filter Parameters command to configure VLAN filters. Parameter 62h indicates update to VLAN Filter, the 2nd parameter indicates which VLAN filter (0 in this case), the last parameter is the VLAN ID (0032h).

Step 5: Configure MDEF[0]

```
Update Manageability Filter Parameters [61, 0, 00000040]
```

Use the Update Manageability Filter Parameters command to update Decision Filters (MDEF) (parameter 61h). This will update MDEF[0], as indicated by the 2nd parameter (0).

MDEF value of 00000040:

Bit 2 [1]– VLAN AND

Step 6: - Enable Filtering

`Receive Enable [A5, AABBCCDDEE, 01020304, 00, 00, 00]`

Using the advanced version Receive Enable command, the first parameter:

Receive Enable Control A5h:

Bit 0 [1] – Enable Receiving of packets

Bit 2 [1] – Enable status reporting (such as link lost)

Bit 5:4 [00]– Notification method = SMB Alert

Bit 6 [1] - Reserved, must be set to 1

Bit 7 [1]– Use Dedicated MAC

Second parameter is the MAC address: AABBCCDDEE.

Third Parameter is the IP address: 01020304.

The last three parameters are zero when the notification method is SMB Alert.

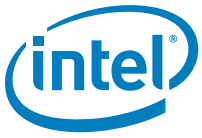


Table 5-19. MDEF Results

Filter		Manageability Decision Filter (MDEF)							
		0	1	2	3	4	5	6	7
L2 Unicast Address	AND								x
Broadcast	AND								
Manageability VLAN	AND	x							
IP Address	AND								
L2 Unicast Address	OR								
Broadcast	OR								
Multicast	AND								
ARP Request	OR								
ARP Response	OR								
Neighbor Discovery	OR								
Port 0x298	OR								
Port 0x26F	OR								
Flex Port 15:0	OR								
Flex TCO 3:0	OR								

5.4 NC-SI Interface

The Network Controller Sideband Interface (NC-SI) is a DMTF industry standard protocol for the sideband interface. NC-SI uses a modified version of the industry standard RMII interface for the physical layer as well as defining a new logical layer.

The NC-SI specification can be found at the DMTF website at:

http://www.dmtf.org/

5.4.1 Overview

5.4.1.1 Terminology

The terminology in this section is taken directly from the NC-SI specification and is as follows:

Term	Definition
Frame Versus Packet	Frame is used in reference to Ethernet, whereas packet is used everywhere else.

External Network Interface	The interface of the network controller that provides connectivity to the external network infrastructure (port).
Internal Host Interface	The interface of the network controller that provides connectivity to the host OS running on the platform.
Management Controller (MC)	An intelligent entity comprising of HW/FW/SW, that resides within a platform and is responsible for some or all management functions associated with the platform (BMC, service processor, etc.).
Network Controller (NC)	The component within a system that is responsible for providing connectivity to the external Ethernet networked world.
Remote Media	The capability to allow remote media devices to appear as if they were attached locally to the host.
Network Controller Sideband Interface	The interface of the network controller that provides connectivity to a management controller. It can be shorten to sideband interface as appropriate in the context.
Term	**Definition**
Interface	This refers to the entire physical interface, such as both the transmit and receive interface between the management controller and the network controller.
Integrated Controller	The term integrated controller refers to a network controller device that supports two or more channels for NC-SI that share a common NC-SI physical interface. For example, a network controller that has two or more physical network ports and a single NC-SI bus connection.
Multi-Drop	Multi-drop commonly refers to the case where multiple physical communication devices share an electrically common bus and a single device acts as the master of the bus and communicates with multiple slave or target devices. In NC-SI, a management controller serves the role as the master, and the network controllers are the target devices.
Point-to-Point	Point-to-point commonly refers to the case where only two physical communication devices are interconnected via a physical communication medium. The devices might be in a master/slave relationship, or could be peers. In NC-SI, point-to-point operation refers to the situation where only a single management controller and single network controller package are used on the bus in a master/slave relationship where the management controller is the master.
Channel	The control logic and data paths supporting NC-SI pass-through operation on a single network interface (port). A network controller that has multiple network interface ports can support an equivalent number of NC-SI channels.
Package	One or more NC-SI channels in a network controller that share a common set of electrical buffers and common buffer control for the NC-SI bus. Typically, there will be a single, logical NC-SI package for a single physical network controller package (chip or module). However, the specification allows a single physical chip or module to hold multiple NC-SI logical packages.
Control Traffic/Messages/Packets	Command, response and notification packets transmitted between MC and NCs for the purpose of managing NC-SI.
Pass-Through Traffic/Messages/ Packets	Non-control packets passed between the external network and the MC through the NC.
Channel Arbitration	Refer to operations where more than one of the network controller channels can be enabled to transmit pass-through packets to the MC at the same time, where arbitration of access to the RXD, CRS_DV, and RX_ER signal lines is accomplished either by software of hardware means.
Logically Enabled/Disabled NC	Refers to the state of the network controller wherein pass-through traffic is able/unable to flow through the sideband interface to and from the management controller, as a result of issuing Enable/Disable Channel command.
NC RX	Defined as the direction of ingress traffic on the external network controller interface

NC TX	Defined as the direction of egress traffic on the external network controller interface
NC-SI RX	Defined as the direction of ingress traffic on the sideband enhanced NC-SI Interface with respect to the network controller.
NC-SI TX	Defined as the direction of egress traffic on the sideband enhanced NC-SI Interface with respect to the network controller.

5.4.1.2 System Topology

In NC-SI each physical endpoint (NC package) can have several logical slaves (NC channels).

NC-SI defines that one management controller and up to four network controller packages can be connected to the same NC-SI link.

Figure 5-10 shows an example topology for a single MC and a single NC package. In this example the NC package has two NC channels.

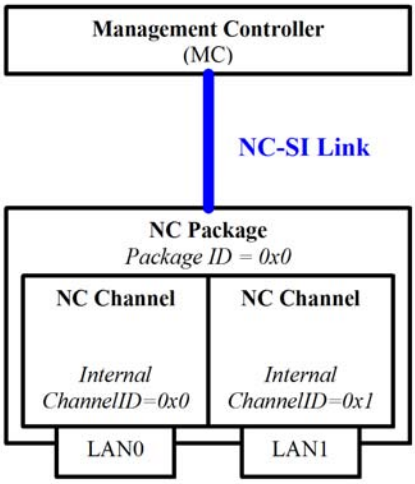


Figure 5-10. Single NC Package, Two NC Channels

Figure 5-11 shows an example topology for a single MC and two NC packages. In this example, one NC package has two NC channels and the other has only one NC channel.

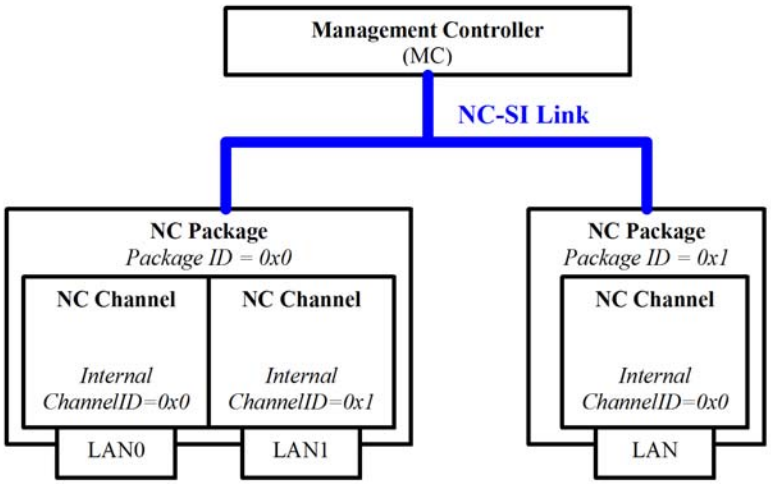


Figure 5-11. Two NC Packages (Left, with Two NC Channels and Right, with One NC Channel)

Scenarios in which the NC-SI lines are shard by multiple NCs (as shown in Figure 5-11) mandate an arbitration mechanism. The arbitration mechanism is described in Section 5.4.5.1.

5.4.1.3 Data Transport

Since NC-SI uses the RMII transport layer, data is transferred in the form of Ethernet frames.

NC-SI defines two types of frames transmitted on the NC-SI interface:

1. Control frames:
   a. Frames used to configure and control the interface.
   b. Control frames are identified by a unique EtherType in their L2 header.
2. Pass-through frames:
   a. The actual LAN pass-through frames transferred from/to the MC.
   b. Pass-through frames are identified as not being a control frame.
   c. Pass-through frames are attributed to a specific NC channel by their source MAC address (as configured in the NC by the MC).

5.4.1.3.1 Control Frames

NC-SI control frames are identified by a unique NC-SI EtherType (0x88F8).

Control frames are used in a single-threaded operation, meaning commands are generated only by the MC and can only be sent one at a time. Each command from the MC is followed by a single response from the NC (command-response flow), after which the MC is allowed to send a new command.

The only exception to the command-response flow is the Asynchronous Event Notification (AEN). These control frames are sent unsolicited from the NC to the MC.

Note: AEN functionality by the NC must be disabled by default, until activated by the MC using the Enable AEN commands.

In order to be considered a valid command, the control frame must:

1. Comply with the NC-SI header format.
2. Be targeted to a valid channel in the package via the *Package ID* and *Channel ID* fields.

For example, to target a NC channel with package ID of 0x2 and internal channel ID of 0x5, The MC must set the channel ID inside the control frame to 0x45.

Note: Channel ID is composed of three bits of package ID and five bits of internal channel ID.

3. Contain a correct payload checksum (if used).
4. Meet any other condition defined by NC-SI.

Note: There are also commands (such as select package) targeted to the package as a whole. These commands must use an internal channel ID of 0x1F.

For more details, refer to the NC-SI specification.

5.4.1.3.2 NC-SI Frames Receive Flow

Figure 5-12 shows the overall flow for frames received on the NC from the MC.

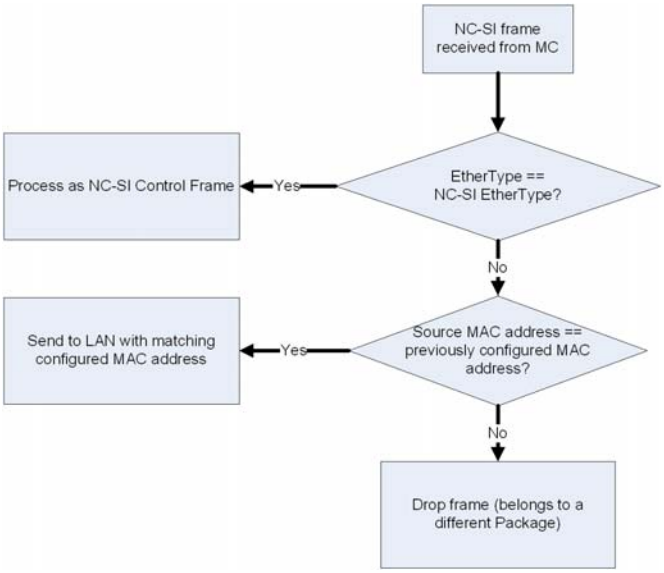


Figure 5-12. NC-SI Frames Receive Flow for the NC

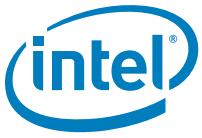
5.4.2 NC-SI Support

5.4.2.1 Supported Features

The 82598 supports the all the mandatory features of the NC-SI specification. Table 5-20 lists the supported commands.

Table 5-20. Supported NC-SI commands

Command	Supported?
Clear Initial State	Yes
Get Version ID	Yes
Get Parameters	Yes
Get Controller Packet Statistics	Yes, partially
Get Link Status	Yes
Enable Channel	Yes
Disable Channel	Yes
Reset Channel	Yes
Enable VLAN	Yes. The 82598 does not support filtering of User priority/CFI Bits of VLAN
Disable VLAN	Yes
Enable Broadcast	Yes
Disable Broadcast	Yes
Set MAC Address	Yes
Clear MAC Address	Yes
Get NC-SI Statistics	Yes, partially
Enable NC-SI Flow-Control	No
Disable NC-SI Flow-Control	No
Set Link Command	Yes
Enable Global Multicast Filter	Yes
Disable Global Multicast Filter	Yes
Get Capabilities	Yes
Set VLAN Filters	Yes
AEN Enable	Yes
Get Pass-Through Statistics	Yes, partially
Select Package	Yes
Deselect Package	Yes
Enable Channel Network TX	Yes
Command	Supported?



Disable Channel Network TX	Yes
OEM Command	Yes

Table 5-21 lists the different capabilities and parameters that are advertised by the 82598 (per each NC-SI channel):

Table 5-21. NC-SI Capabilities Advertisement

Feature	Capability	Details
Capabilities Flags	Hardware arbitration	Supported
	OS presence	Supported
	Network controller to management controller flow control support	Not supported
	Management Controller to Network Controller Flow Control Support	Not supported
	All multicast addresses support	Supported
Broadcast Filtering	ARP	Supported
	DHCP client	Supported
	DHCP Server	Supported
	NetBIOS	Supported
Multicast Filtering	IPv6 neighbor advertisement	Supported
	IPv6 router advertisement	Supported
	DHCPv6 relay and server multicast	Supported
Buffering Capabilities	LAN receive buffer size	8Kbytes
AEN Support		All AENs are supported
MAC Filtering	Unicast filters	0
	Multicast filters	0
	Mixed filters	2
VLAN Filtering	VLAN filters	8
VLAN Mode Support	Supported modes	Modes 1 and 3
Feature	**Capability**	**Details**
Channel Count	Number of channels in the package	2 channels

Version ID	NC-SI version	1.0
	FW name	The 82598 ROM Verified
	FW version	0x20D0002
	PCI DID	0x10A7
	PCI VID	0x8086
	Manufacturer ID (IANA)	0x157 (Intel)

Table 5-22 lists the optional features supported:

Table 5-22. Optional NC-SI Features Support

Feature	Supported	Details
AENs	Yes	**Note:** The driver state AEN might be emitted up to 15 seconds after actual driver change.
Get Controller Packet Statistics Command	No	
Get NC-SI Statistics Command	Yes, partially	Supports the following counters: 1-4, 7.
Get NC-SI Pass-Through Statistics Command	Yes, partially	Supports the following counters: 2 Support the following counters only when the OS is down: 1, 6, 7.
VLAN Modes	Yes, partially	Supports only modes 1, 3.
MAC Address Filters	Yes	Supports two MAC addresses as mixed per port.
Channel Count	Yes	Supports two channels.
VLAN Filters	Yes	Supports eight VLAN filters per port.
Broadcast Filters	Yes	Supports the following filters: • ARP. • DHCP. • NetBIOS.
Multicast Filters	Yes	Supports the following filters: • IPv6 neighbor advertisement. • IPv6 router advertisement. • DHCPv6 relay and server multicast.
NC-SI Flow Control Command	No	Does not support NC-SI flow-control.
Feature	**Supported**	**Details**
HW Arbitration	No	Does not support NC-SI HW arbitration.
Allow Link Down	No	Does not support shutting down the link when disabled.

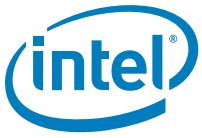
5.4.2.2 NC-SI Mode - Intel Specific Commands

In addition to the regular NC-SI commands, the following Intel vendor specific commands are supported. The purpose of these commands is to provide a means for the Baseboard Management Controller (BMC) to access some of the Intel-specific features present in the 82598.

5.4.2.2.1 Overview

The following features are available via the NC-SI OEM specific command:

- Receive filters:
  - Packet Addition Decision Filters 0x0…0x4
  - Packet Reduction Decision Filters 0x5…0x7
  - MNG2HOST register (controls the forwarding of manageability packets to the host)



  - — Flex 128 filters 0x0…0x3
  - — Flex TCP/UDP port filters 0x0...0xA
  - — IPv4/IPv6 filters
- Get System MAC Address - This command allows the MC to retrieve the system MAC address used by the NC. This MAC address can be used for a shared MAC address mode.
- Keep Phy Link Up (*Veto* bit) Enable/Disable - This feature enables the BMC to block Phy reset, which might cause session loss.
- TCO Reset - Allows the MC to reset the 82598.
- Checksum offloading - Offloads IP/UDP/TCP checksum checking from the MC.

These commands are designed to be compliant with their corresponding SMBus commands (if existing).

All of the commands are based on a single DMTF defined NC-SI command, known as OEM Command. This command is as follows.

5.4.2.2.1.1 OEM Command (0x50)

The OEM command can be used by the MC to request the sideband interface to provide vendor-specific information. The Vendor Enterprise Number (VEN) is the unique MIB/SNMP private enterprise number assigned by IANA per organization. Vendors are free to define their own internal data structures in the vendor data fields.

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..	Intel Command Number	Optional Data		

Figure 5-13. OEM Command Packet Format

5.4.2.2.1.2 OEM Response (0xD0)

Below is the vendor specific format for commands, as defined by NC-SI.

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	Intel Command Number	Optional Return Data		

Figure 5-14. OEM Response Packet Format

5.4.2.2.1.3 OEM Specific Command Response Reason Codes

Response Code		Reason Code	
Value	Description	Value	Description
0x1	Command Failed	0x5081	Invalid Intel Command Number
0x1	Command Failed	0x5082	Invalid Intel Command Parameter Number
0x1	Command Failed	0x5085	Internal Network Controller Error
0x1	Command Failed	0x5086	Invalid Vendor Enterprise Code

Table 5-23. Commands Summary

Intel Command	Parameter	Command Name
0x00	0x00	Set IP Filters Control
0x01	0x00	Get IP Filters Control
0x02	0x0A	Set Manageability to Host
	0x10	Set Flexible 128 Filter 0 Mask and Length
	0x11	Set Flexible 128 Filter 0 Data
	0x20	Set Flexible 128 Filter 1 Mask and Length
	0x21	Set Flexible 128 Filter 1 Data
	0x30	Set Flexible 128 Filter 2 Mask and Length
	0x31	Set Flexible 128 Filter 2 Data
	0x40	Set Flexible 128 Filter 3 Mask and Length
	0x41	Set Flexible 128 Filter 3 Data
	0x61	Set Packet Addition Filters
	0x63	Set Flex TCP/UDP Port Filters
	0x64	Set Flex IPv4 Address Filters
	0x65	Set Flex IPv6 Address Filters
0x3	0x0A	Get Manageability to Host
	0x10	Get Flexible 128 Filter 0 Mask and Length
	0x11	Get Flexible 128 Filter 0 Data
	0x20	Get Flexible 128 Filter 1 Mask and Length
	0x21	Get Flexible 128 Filter 1 Data

	0x30	Get Flexible 128 Filter 2 Mask and Length
	0x31	Get Flexible 128 Filter 2 Data
	0x40	Get Flexible 128 Filter 3 Mask and Length
	0x41	Get Flexible 128 Filter 3 Data
	0x61	Get Packet Addition Filters
	0x63	Get Flex TCP/UDP Port Filters
	0x64	Get Flex IPv4 Address Filters
	0x65	Get Flex IPv6 Address Filters
Intel Command	**Parameter**	**Command Name**
0x04	0x00	Set Unicast Packet Reduction
	0x01	Set Multicast Packet Reduction
	0x02	Set Broadcast Packet Reduction
0x05	0x00	Get Unicast Packet Reduction
	0x01	Get Multicast Packet Reduction
	0x02	Get Broadcast Packet Reduction
0x06	N/A	Get System MAC Address
0x20	N/A	Set Intel Management Control
0x21	N/A	Get Intel Management Control
0x22	N/A	Perform TCO Reset
0x23	N/A	Enable IP/UDP/TCP Checksum Offloading
0x24	N/A	Disable IP/UDP/TCP Checksum Offloading

5.4.2.2.2 Proprietary Commands Format

5.4.2.2.2.1 Set Intel Filters Control Command (Intel Command 0x00)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x00	Filter Control Index		

5.4.2.2.2 Set Intel Filters Control Response Format (Intel Command 0x00)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x00	Filter Control Index		

5.4.2.2.3 Set Intel Filters Control - IP Filters Control Command (Intel Command 0x00, Filter Control Index 0x00)

This command controls different aspects of the Intel filters.

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x00	0x00	IP Filters control (3-2)	
24..27	IP Filters Control (1-0)			

Where "IP Filters Control" has the following format:

Bit #	Name	Description	Default Value
0	IPv4/IPv6 Mode	IPv6 (0b): There are zero IPv4 filters and four IPv6 filters IPv4 (1b): There are four IPv4 filters and three IPv6 filters	1b
1..15	Reserved		
16	IPv4 Filter 0 Valid	Indicates if the IPv4 address configured in IPv4 address 0 is valid.	0b **Note:** The network controller automatically sets this bit to 1b if the Set Intel Filter – IPv4 Filter Command is used for filter zero.
17	IPv4 Filter 1 Valid	Indicates if the IPv4 address configured in IPv4 address 1 is valid.	0b **Note:** The network controller automatically sets this bit to 1b if the Set Intel Filter – IPv4 Filter Command is used for filter one.
18	IPv4 Filter 2 Valid	Indicates if the IPv4 address configured in IPv4 address 2 is valid.	0b **Note:** The network controller automatically sets this bit to 1b if the Set Intel Filter – IPv4 Filter Command is used for filter two.

19	IPv4 Filter 3 Valid	Indicates if the IPv4 address configured in IPv4 address 3 is valid.	0b **Note:** The network controller automatically sets this bit to 1b if the Set Intel Filter – IPv4 Filter Command is used for filter three.
20..23	Reserved		
24	IPv6 Filter 0 Valid	Indicates if the IPv6 address configured in IPv6 address 0 is valid.	0b **Note:** The network controller automatically sets this bit to 1b if the Set Intel Filter – IPv6 Filter Command is used for filter zero.
Bit #	**Name**	**Description**	**Default Value**
25	IPv6 Filter 1 Valid	Indicates if the IPv6 address configured in IPv6 address 1 is valid.	0b **Note:** The network controller automatically sets this bit to 1b if the Set Intel Filter – IPv6 Filter Command is used for filter one.
26	IPv6 Filter 2 Valid	Indicates if the IPv6 address configured in IPv6 address 2 is valid.	0b **Note:** The network controller automatically sets this bit to 1b if the Set Intel Filter – IPv6 Filter Command is used for filter two.
27	IPv6 Filter 3 Valid	Indicates if the IPv6 address configured in IPv6 address 3 is valid.	0b **Note:** The network controller automatically sets this bit to 1b if the Set Intel Filter – IPv6 Filter Command is used for filter three.
28..31	Reserved		

5.4.2.2.3.1 Set Intel Filters Control - IP Filters Control Response (Intel Command 0x00, Filter Control Index 0x00)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x00	0x00		

5.4.2.2.4 Get Intel Filters Control Command (Intel Command 0x01)

5.4.2.2.4.1 Get Intel Filters Control - IP Filters Control Command (Intel Command 0x01, Filter Control Index 0x00)

This command controls different aspects of the Intel filters.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x01	0x00		

5.4.2.2.4.2 Get Intel Filters Control - IP Filters Control Response (Intel Command 0x01, Filter Control Index 0x00)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x01	0x00	IP Filters Control (3-2)	
28..29	IP Filters Control (1-0)			

5.4.2.2.5 Set Intel Filters Formats

5.4.2.2.5.1 Set Intel Filters Command (Intel Command 0x02)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x02	Parameter Number	Filters Data (optional)	

5.4.2.2.5.2 Set Intel Filters Response (Intel Command 0x02)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..	0x02	Filter Control Index	Return Data (Optional)	

5.4.2.2.5.3 Set Intel Filters - Manageability to Host Command (Intel Command 0x02, Filter Parameter 0x0A)

This command sets the Mng2Host register. The Mng2Host register controls whether pass-through packets destined to the BMC are also be forwarded to the host OS.

The Mng2Host register has the following structure:

Bits	Description	Default
0	Decision Filter 0	Determines if packets that have passed Decision Filter 0 is also forwarded to the host OS.
1	Decision Filter 1	Determines if packets that have passed Decision Filter 1 is also forwarded to the host OS.
2	Decision Filter 2	Determines if packets that have passed Decision Filter 2 is also forwarded to the host OS.
3	Decision Filter 3	Determines if packets that have passed Decision Filter 3 is also forwarded to the host OS.
4	Decision Filter 4	Determines if packets that have passed Decision Filter 4 is also forwarded to the host OS.
5	Unicast and Mixed	Determines if broadcast packets are also forwarded to the host OS.
6	Global Multicast	Determines if unicast and mixed packets are also forwarded to the host OS.
7	Broadcast	Determines if global multicast packets are also forwarded to the host OS.

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x02	0x0A	Manageability to Host (3-2)	
24..25	Manageability to Host (1-0)			

5.4.2.2.5.4 Set Intel Filters - Manageability to Host Response (Intel Command 0x02, Filter Parameter 0x0A)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..	0x02	0x0A		

5.4.2.2.5.5 Set Intel Filters - Flex Filter 0 Enable Mask and Length Command (Intel Command 0x02, Filter Parameter 0x10/0x20/0x30/0x40)

The following command sets the Intel flex filters mask and length. Use filter parameters 0x10/0x20/ 0x30/0x40 for flexible filters 0/1/2/3 accordingly.

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x02	0x10/ 0x20/ 0x30/ 0x40	Mask Byte 1	Mask Byte 2
24..27	..	..	..	..
28...31	..	..	..	..
32...35	..	..	..	..
35...37	..	Mask Byte 16	Reserved	Reserved
38	Flexible Filter Length			

5.4.2.2.5.6 Set Intel Filters - Flex Filter 0 Enable Mask and Length Response (Intel Command 0x02, Filter Parameter 0x10/0x20/0x30/0x40)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x02	0x10/ 0x20/ 0x30/ 0x40		

5.4.2.2.5.7 Set Intel Filters - Flex Filter 0 Data Command (Intel Command 0x02, Filter Parameter 0x11/0x21/0x31/0x41)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..	0x02	0x11/ 0x21/ 0x31/ 0x41	Filter Data Group	Filter Data 1
	..	Filter Data N		

Note: Using this command to configure the filters data must be done after the flex filter mask command is issued and the mask is set.

5.4.2.2.5.8 Set Intel Filters - Flex Filter 0 Data Response (Intel Command 0x02, Filter Parameter 0x11/0x21/0x31/0x41)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x02	0x11/ 0x21/ 0x31/ 0x41		

5.4.2.2.5.9 Set Intel Filters - Packet Addition Decision Filter Command (Intel Command 0x02, Filter Parameter 0x61)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x02	0x61	Decision Filter (3-2)	
24..25	Decision Filter (1-0)			

Filter index range: 0x0..0x4.

Bit #	Name	Description
0	Unicast (AND)	If set, packets must match a unicast filter.
1	Broadcast (AND)	If set, packets must match the broadcast filter.
2	VLAN (AND)	If set, packets must match a VLAN filter.
3	IP Address (AND)	If set, packets must match an IP filter.
4	Unicast (OR)	If set, packets must match a unicast filter or a different OR filter.
5	Broadcast	If set, packets must match the broadcast filter or a different OR filter.
6	Multicast (AND)	If set, packets must match the multicast filter.
7	ARP Request (OR)	If set, packets must match the ARP request filter or a different OR filter.
8	ARP Response (OR)	If set, packets must also match the ARP response filter or a different OR filter.
9	Neighbor Discovery (OR)	If set, packets must also match the neighbor discovery filter or a different OR filter.
10	Port 0x298 (OR)	If set, packets must also match a fixed TCP/UDP port 0x298 filter or a different OR filter.
11	Port 0x26F (OR)	If set, packets must also match a fixed TCP/UDP port 0x26F filter or a different OR filter.
12	Flex port 0 (OR)	If set, packets must also match the TCP/UDP port filter 0 or a different OR filter.
13	Flex port 1 (OR)	If set, packets must also match the TCP/UDP port filter 1 or a different OR filter.
14	Flex port 2 (OR)	If set, packets must also match the TCP/UDP port filter 2 or a different OR filter.
15	Flex port 3 (OR)	If set, packets must also match the TCP/UDP port filter 3 or a different OR filter.
16	Flex port 4 (OR)	If set, packets must also match the TCP/UDP port filter 4 or a different OR filter.
17	Flex port 5 (OR)	If set, packets must also match the TCP/UDP port filter 5 or a different OR filter.
18	Flex port 6 (OR)	If set, packets must also match the TCP/UDP port filter 6 or a different OR filter.
19	Flex port 7 (OR)	If set, packets must also match the TCP/UDP port filter 7 or a different OR filter.
20	Flex port 8 (OR)	If set, packets must also match the TCP/UDP port filter 8 or a different OR filter.
Bit #	**Name**	**Description**
21	Flex port 9 (OR)	If set, packets must also match the TCP/UDP port filter 9 or a different OR filter.
22	Flex port 10 (OR)	If set, packets must also match the TCP/UDP port filter 10 or a different OR filter.
23	Flex port 11 (OR)	If set, packets must also match the TCP/UDP port filter 11 or a different OR filter.
24	Reserved	
25	Reserved	

26	Reserved	
27	Reserved	
28	Flex TCO 0 (OR)	If set, packets must also match the Flex 128 TCO filter 0 or a different OR filter.
29	Flex TCO 1 (OR)	If set, packets must also match the Flex 128 TCO filter 1 or a different OR filter.
30	Flex TCO 2 (OR)	If set, packets must also match the Flex 128 TCO filter 2 or a different OR filter.
31	Flex TCO 3 (OR)	If set, packets must also match the Flex 128 TCO filter 3 or a different OR filter.

The filtering is divided into two decisions:

- Bits 0, 1, 2, 3, and 6 work in an AND manner; they all must be true in order for a packet to pass (if any were set).
- Bits 5 and 7-31 work in an OR manner; at least one of them must be true for a packet to pass (if any were set).

See Figure 5-5 for more details on the decision filters.

Note: These filter settings operate according to the VLAN mode, as configured according to the DMTF NC-SI specification. After disabling packet reduction filters, the MC must re-set the VLAN mode using the Set VLAN command.

5.4.2.2.5.10 Set Intel Filters - Packet Addition Decision Filter Response (Intel Command 0x02, Filter Parameter 0x61)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x02	0x61		

5.4.2.2.5.11 Set Intel Filters - Flex TCP/UDP Port Filter Command (Intel Command 0x02, Filter Parameter 0x63)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x02	0x63	TCP/UDP Port	

Filter index range: 0x0..0xA.

5.4.2.2.5.12 Set Intel Filters - Flex TCP/UDP Port Filter Response (Intel Command 0x02, Filter Parameter 0x63)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x02	0x63		

5.4.2.2.5.13 Set Intel Filters - IPv4 Filter Command (Intel Command 0x02, Filter Parameter 0x64)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x02	0x64	IPv4 Address (3-2)	
24..25	IPv4 Address (3-2)			

Note: The filters index range can vary according to the IPv4/IPv6 mode setting in the Filters Control command.

IPv4 Mode: Filter index range: 0x0..0x3.

IPv6 Mode: No IPv4 Filters.

5.4.2.2.5.14 Set Intel Filters - IPv4 Filter Response (Intel Command 0x02, Filter Parameter 0x64)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x02	0x64		

5.4.2.2.5.15 Set Intel Filters - IPv6 Filter Command (Intel Command 0x02, Filter Parameter 0x65)

Bits / Bytes	31..24	23..16	15..08	07..00
00..03	MC ID	*0x01*	Reserved	IID
04..07	*0x50*	ID	Resvd.	*0x09*
08..11	Reserved			
12..15	Reserved			
16..19	0x8086		0x02	0x65
20...23	IPv6 Filter Index	IPv6 Address (MSB, Byte 15)	..	..
24...27	..	..	..	..
28...31	..	..	..	..
32...35	..	..	..	..
36	IPv6 Address (LSB, Byte 0)			

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x02	0x65	IPv6 Address (MSB, byte 15)	..
24..27	..	..	..	..
28..31	..	..	..	..
32..35	..	..	..	..
36..37	..	IPv6 Address (LSB, byte 0)		

Note: The filters index range can vary according to the IPv4/IPv6 mode setting in the Filters Control command.

IPv4 Mode: Filter index range: 0x0..0x2.

IPv6 Mode: Filter index range: 0x0..0x3.

5.4.2.2.5.16 Set Intel Filters - IPv6 Filter Response (Intel Command 0x02, Filter Parameter 0x65)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x02	0x65		

5.4.2.2.6 Get Intel Filters Formats

5.4.2.2.6.1 Get Intel Filters Command (Intel Command 0x03)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x03	Parameter Number		

5.4.2.2.6.2 Get Intel Filters Response (Intel Command 0x03)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x03	Parameter Number	Optional Return Data	

5.4.2.2.6.3 Get Intel Filters - Manageability to Host Command (Intel Command 0x03, Filter Parameter 0x0A)

This command retrieves the Mng2Host register. The Mng2Host register controls whether pass-through packets destined to the BMC are also be forwarded to the host OS.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x03	0x0A		

5.4.2.2.6.4 Get Intel Filters - Manageability to Host Response (Intel Command 0x03, Filter Parameter 0x0A)

The Mng2Host register has the following structure:

Bits	Description	Default
0	Decision Filter 0	Determines if packets that have passed decision filter 0 are also forwarded to the host OS.
1	Decision Filter 1	Determines if packets that have passed decision filter 1 are also forwarded to the host OS.
2	Decision Filter 2	Determines if packets that have passed decision filter 2 are also forwarded to the host OS.
3	Decision Filter 3	Determines if packets that have passed decision filter 3 are also forwarded to the host OS.
4	Decision Filter 4	Determines if packets that have passed decision filter 4 are also forwarded to the host OS.
5	Unicast and Mixed	Determines if broadcast packets are also forwarded to the host OS.
6	Global Multicast	Determines if unicast packets are also forwarded to the host OS.
7	Broadcast	Determines if multicast packets are also forwarded to the host OS.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x03	0x0A	Manageability to Host (3-2)	
28..29	Manageability to Host (1-0)			

5.4.2.2.6.5 Get Intel Filters - Flex Filter 0 Enable Mask and Length Command (Intel Command 0x03, Filter Parameter 0x10/0x20/0x30/0x40)

The following command retrieves the Intel flex filters mask and length. Use filter parameters 0x10/ 0x20/0x30/0x40 for flexible filters 0/1/2/3 accordingly.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x03	0x10/ 0x20/ 0x30/ 0x40		

5.4.2.2.6.6 Get Intel Filters - Flex Filter 0 Enable Mask and Length Response (Intel Command 0x03, Filter Parameter 0x10/0x20/0x30/0x40)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x03	0x10/ 0x20/ 0x30/ 0x40	Mask Byte 1	Mask Byte 2
28..31	..	..	..	..
32..35	..	..	..	..
36..39	..	..	..	..
40..43	..	Mask Byte 16	Reserved	Reserved
44	Flexible Filter Length			

5.4.2.2.6.7 Get Intel Filters - Flex Filter 0 Data Command (Intel Command 0x03, Filter Parameter 0x11/0x21/0x31/0x41)

The following command retrieves the Intel flex filters data. Use filter parameters 0x11/0x21/0x31/0x41 for flexible filters 0/1/2/3 accordingly.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x03	0x11/ 0x21/ 0x31/ 0x41		

5.4.2.2.6.8 Get Intel Filters - Flex Filter 0 Data Response (Intel Command 0x03, Filter Parameter 0x11)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..	0x03	0x11/ 0x21/ 0x31/ 0x41	Filter Group Number	Filter Data 1
	..	Filter Data N		

5.4.2.2.6.9 Get Intel Filters - Packet Addition Decision Filter Command (Intel Command 0x03, Filter Parameter 0x61)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x03	0x61		

Filter index range: 0x0..0x4.

5.4.2.2.6.10 Get Intel Filters - Packet Addition Decision Filter Response (Intel Command 0x03, Filter Parameter 0x0A)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x03	0x61	Decision Filter (3-2)	
28..29	Decision Filter (1-0)			

5.4.2.2.6.11 Get Intel Filters - Flex TCP/UDP Port Filter Command (Intel Command 0x03, Filter Parameter 0x63)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..22	0x03	0x63	TCP/UDP Filter Index	

Filter index range: 0x0..0xA.

5.4.2.2.6.12 Get Intel Filters - Flex TCP/UDP Port Filter Response (Intel Command 0x03, Filter Parameter 0x63)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x03	0x63	TCP/UDP Filter Index	TCP/UDP Port (1)
28	TCP/UDP Port (0)			

Filter index range: 0x0..0xA.

5.4.2.2.6.13 Get Intel Filters - IPv4 Filter Command (Intel Command 0x03, Filter Parameter 0x64)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..22	0x03	0x64	IPv4 Filter Index	

Note: The filters index range can vary according to the IPv4/IPv6 mode setting in the Filters Control command.

IPv4 Mode: Filter index range: 0x0..0x3.

IPv6 Mode: No IPv4 filters.

5.4.2.2.6.14 Get Intel Filters - IPv4 Filter Response (Intel Command 0x03, Filter Parameter 0x64)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x03	0x64	IPv4 Filter Index	IPv4 Address (3)
28..29	IPv4 Address (2-0)			

5.4.2.2.6.15 Get Intel Filters - IPv6 Filter Command (Intel Command 0x03, Filter Parameter 0x65)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..22	0x03	0x65	IPv6 Filter Index	

Note: The filters index range can vary according to the IPv4/IPv6 mode setting in the Filters Control command

IPv4 Mode: Filter index range: 0x0..0x2.

IPv6 Mode: Filter index range: 0x0..0x3.

5.4.2.2.6.16 Get Intel Filters - IPv6 Filter Response (Intel Command 0x03, Filter parameter 0x65)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x03	0x65	IPv6 Filter Index	IPv6 Address (MSB, Byte 16)
28..31	..	..	..	..
32..35	..	..	..	..
36..39	..	..	..	..
40..42	..	..	IPv6 Address (LSB, Byte 0)	

5.4.2.2.7 Set Intel Packet Reduction Filters Formats

5.4.2.2.7.1 Set Intel Packet Reduction Filters Command (Intel Command 0x04)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x04	Packet Reduction Index		

5.4.2.2.7.2 Set Intel Packet Reduction Filters Response (Intel Command 0x04)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..	0x04	Packet Reduction Index	Optional Return Data	

5.4.2.2.7.3 Set Unicast Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x00)

This command causes the NC to filter packets that have passed due to the unicast filter (MAC address filters, as specified in the DMTF NC-SI). Note that unicast filtering might be affected by other filters, as specified in the DMTF NC-SI.

The filtering of these packets are done such that the MC might add a logical condition that a packet must match, or it must be discarded.

Note: Packets that might have been blocked can still pass due to other decision filters.

In order to disable unicast packet reduction, the MC should set all reduction filters to 0b. Following such a setting the NC must forward, to the MC, all packets that have passed the unicast filters (MAC address filtering) as specified in the DMTF NC-SI.

The *Unicast Packet Reduction* field has the following structure:

Bit #	Name	Description
0	Reserved	
1	Reserved	
2	Reserved	
3	IP Address	If set, all unicast packets must also match an IP filter.
4	Reserved	
5	Reserved	
6	Reserved	
7	Reserved	
8	ARP Response	If set, all unicast packets must also match the ARP response filter (any of the active filters).
9	Reserved	
10	Port 0x298	If set, all unicast packets must also match a fixed TCP/UDP port 0x298 filter.
11	Port 0x26F	If set, all unicast packets must also match a fixed TCP/UDP port 0x26F filter.
12	Flex port 0	If set, all unicast packets must also match the TCP/UDP port filter 0.
13	Flex port 1	If set, all unicast packets must also match the TCP/UDP port filter 1.
14	Flex port 2	If set, all unicast packets must also match the TCP/UDP port filter 2.
15	Flex port 3	If set, all unicast packets must also match the TCP/UDP port filter 3.
16	Flex port 4	If set, all unicast packets must also match the TCP/UDP port filter 4.
17	Flex port 5	If set, all unicast packets must also match the TCP/UDP port filter 5.
18	Flex port 6	If set, all unicast packets must also match the TCP/UDP port filter 6.
Bit #	Name	Description

19	Flex port 7	If set, all unicast packets must also match the TCP/UDP port filter 7.
20	Flex port 8	If set, all unicast packets must also match the TCP/UDP port filter 8.
21	Flex port 9	If set, all unicast packets must also match the TCP/UDP port filter 9.
22	Flex port 10	If set, all unicast packets must also match the TCP/UDP port filter 10.
23	Reserved	
24	Reserved	
25	Reserved	
26	Reserved	
27	Reserved	
28	Flex TCO 0	If set, all unicast packets must also match the flex 128 TCO filter 0.
29	Flex TCO 1	If set, all unicast packets must also match the flex 128 TCO filter 1.
30	Flex TCO 2	If set, all unicast packets must also match the flex 128 TCO filter 2.
31	Flex TCO 3	If set, all unicast packets must also match the flex 128 TCO filter 3.

The filtering is divided into two decisions:

- Bit 3 works in an AND manner; it must be true in order for a packet to pass (if was set).
- Bits 8-31 work in an OR manner; at least one of them must be true for a packet to pass (if any were set).

See Figure 5-5 for more details on the decision filters.

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x04	0x00	Unicast Packet Reduction (3-2)	
24..25	Unicast Packet Reduction (1-0)			

5.4.2.2.7.4 Set Unicast Packet Reduction Response (Intel Command 0x04, Reduction Filter Index 0x00)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x04	0x00		

5.4.2.2.7.5 Set Multicast Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x01)

This command causes the NC to filter packets that have passed due to the multicast filter (MAC address filters, as specified in the DMTF NC-SI).

The filtering of these packets are done such that the MC might add a logical condition that a packet must match, or it must be discarded.

Note: Packets that might have been blocked can still pass due to other decision filters.

In order to disable mulitcast packet reduction, the MC should set all reduction filters to 0b. Following such a setting, the NC must forward, to the MC, all packets that have passed the multicast filters (global multicast filtering) as specified in the DMTF NC-SI.

The *Multicast Packet Reduction* field has the following structure:

Bit #	Name	Description
0	Reserved	Reserved.
1	Reserved	
2	Reserved	
3	IP Address	If set, all multicast packets must also match an IP filter.
4	Reserved	
5	Reserved	
6	Reserved	
7	Reserved	
8	ARP Response	If set, all multicast packets must also match the ARP response filter (any of the active filters).
9	Reserved	
10	Port 0x298	If set, all multicast packets must also match a fixed TCP/UDP port 0x298 filter.
Bit #	**Name**	**Description**

11	Port 0x26F	If set, all multicast packets must also match a fixed TCP/UDP port 0x26F filter.
12	Flex port 0	If set, all multicast packets must also match the TCP/UDP port filter 0.
13	Flex port 1	If set, all multicast packets must also match the TCP/UDP port filter 1.
14	Flex port 2	If set, all multicast packets must also match the TCP/UDP port filter 2.
15	Flex port 3	If set, all multicast packets must also match the TCP/UDP port filter 3.
16	Flex port 4	If set, all multicast packets must also match the TCP/UDP port filter 4.
17	Flex port 5	If set, all multicast packets must also match the TCP/UDP port filter 5.
18	Flex port 6	If set, all multicast packets must also match the TCP/UDP port filter 6.
19	Flex port 7	If set, all multicast packets must also match the TCP/UDP port filter 7.
20	Flex port 8	If set, all multicast packets must also match the TCP/UDP port filter 8.
21	Flex port 9	If set, all multicast packets must also match the TCP/UDP port filter 9.
22	Flex port 10	If set, all multicast packets must also match the TCP/UDP port filter 10.
23	Reserved	
24	Reserved	
25	Reserved	
26	Reserved	
27	Reserved	
28	Flex TCO 0	If set, all multicast packets must also match the flex 128 TCO filter 0.
29	Flex TCO 0	If set, all multicast packets must also match the flex 128 TCO filter 1.
30	Flex TCO 0	If set, all multicast packets must also match the flex 128 TCO filter 2.
31	Flex TCO 0	If set, all multicast packets must also match the flex 128 TCO filter 3.

The filtering is divided into two decisions:

Bit 3 works in an AND manner; it must be true in order for a packet to pass (if was set).

Bits 4, 5, and 7-31 work in an OR manner; at least one of them must be true for a packet to pass (if any were set).

See Figure 5-5 for more details on the decision filters.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x04	0x01	Multicast Packet Reduction (3-2)	
24..25	Multicast Packet Reduction (1-0)			

5.4.2.2.7.6 Set Multicast Packet Reduction Response (Intel Command 0x04, Reduction Filter Index 0x01)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x04	0x01		

5.4.2.2.7.7 Set Broadcast Packet Reduction Command (Intel Command 0x04, Reduction Filter Index 0x02)

This command causes the NC to filter packets that have passed due to the broadcast filter (MAC address filters, as specified in the DMTF NC-SI).

The filtering of these packets are done such that the MC might add a logical condition that a packet must match, or it must be discarded.

Note: Packets that might have been blocked can still pass due to other decision filters.

In order to disable broadcast packet reduction, the MC should set all reduction filters to 0b. Following such a setting, the NC must forward, to the MC, all packets that have passed the broadcast filters as specified in the DMTF NC-SI.

The *Broadcast Packet Reduction* field has the following structure:

Bit #	Name	Description
0	Reserved	Reserved.
1	Reserved	
2	Reserved	
3	IP Address	If set, all broadcast packets must also match an IP filter.
4	Reserved	

5	Reserved	
6	Reserved	
7	Reserved	
8	ARP Response	If set, all broadcast packets must also match the ARP response filter (any of the active filters).
9	Reserved	
10	Port 0x298	If set, all broadcast packets must also match a fixed TCP/UDP port 0x298 filter.
11	Port 0x26F	If set, all broadcast packets must also match a fixed TCP/UDP port 0x26F filter.
12	Flex port 0	If set, all broadcast packets must also match the TCP/UDP port filter 0.
13	Flex port 1	If set, all broadcast packets must also match the TCP/UDP port filter 1.
14	Flex port 2	If set, all broadcast packets must also match the TCP/UDP port filter 2.
15	Flex port 3	If set, all broadcast packets must also match the TCP/UDP port filter 3.
16	Flex port 4	If set, all broadcast packets must also match the TCP/UDP port filter 4.
17	Flex port 5	If set, all broadcast packets must also match the TCP/UDP port filter 5.
18	Flex port 6	If set, all broadcast packets must also match the TCP/UDP port filter 6.
19	Flex port 7	If set, all broadcast packets must also match the TCP/UDP port filter 7.
20	Flex port 8	If set, all broadcast packets must also match the TCP/UDP port filter 8.
21	Flex port 9	If set, all broadcast packets must also match the TCP/UDP port filter 9.
22	Flex port 10	If set, all broadcast packets must also match the TCP/UDP port filter 10.
23	Reserved	
24	Reserved	
25	Reserved	
26	Reserved	
27	Reserved	
28	Flex TCO 0	If set, all broadcast packets must also match the flex 128 TCO filter 0.
29	Flex TCO 0	If set, all broadcast packets must also match the flex 128 TCO filter 1.
30	Flex TCO 0	If set, all broadcast packets must also match the flex 128 TCO filter 2.
31	Flex TCO 0	If set, all broadcast packets must also match the flex 128 TCO filter 3.

The filtering is divided into two decisions:

Bit 3 works in an AND manner; it must be true in order for a packet to pass (if was set).

Bits 4, 5, and 7-31 work in an OR manner; at least one of them must be true for a packet to pass (if any were set).

See Figure 5-5 for more details on the decision filters.

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..23	0x04	0x02	Broadcast Packet Reduction (3-2)	
24..25	Broadcast Packet Reduction (1-0)			

5.4.2.2.7.8 Set Broadcast Packet Reduction Response (Intel Command 0x08)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x04	0x02		

5.4.2.2.8 Get Intel Packet Reduction Filters Formats

5.4.2.2.8.1 Get Intel Packet Reduction Filters Command (Intel Command 0x05)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x05	Reduction Filter Index		

5.4.2.2.8.2 Set Intel Packet Reduction Filters Response (Intel Command 0x05)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..	0x05	Reduction Filter Index	Optional Return Data	

5.4.2.2.8.3 Get Unicast Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x00)

This command causes the NC to disable any packet reductions for unicast address filtering.

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x05	0x00		

5.4.2.2.8.4 Get Unicast Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x00)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x05	0x00	Unicast Packet Reduction (3-2)	
28..29	Unicast Packet Reduction (1-0)			

5.4.2.2.8.5 Get Multicast Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x01)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x05	0x01		

5.4.2.2.8.6 Get Multicast Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x01)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x05	0x00	Multicast Packet Reduction (3-2)	
28..29	Multicast Packet Reduction (1-0)			

5.4.2.2.8.7 Get Broadcast Packet Reduction Command (Intel Command 0x05, Reduction Filter Index 0x02)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x05	0x02		

5.4.2.2.8.8 Get Broadcast Packet Reduction Response (Intel Command 0x05, Reduction Filter Index 0x02)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x05	0x00	Broadcast Packet Reduction (3-2)	
28..29	Broadcast Packet Reduction (1-0)			

5.4.2.2.9 System MAC Address

5.4.2.2.9.1 Get System MAC Address Command (Intel Command 0x06)

In order to support a system configuration that requires the NC to hold the MAC address for the MC (such as shared MAC address mode), the following command is provided to enable the MC to query the NC for a valid MAC address.

The NC must return the system MAC addresses. The MC should use the returned MAC addressing as a shared MAC address by setting it using the Set MAC Address command as defined in NC-SI 1.0.

It is also recommended that the MC use packet reduction and Manageability-to-Host command to set the proper filtering method.

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20	0x06			

5.4.2.2.9.2 Get System MAC Address Response (Intel Command 0x06)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..27	0x06	MAC Address		
28..30	MAC Address			

5.4.2.2.10 Set Intel Management Control Formats

5.4.2.2.10.1 Set Intel Management Control Command (Intel Command 0x20)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..22	0x20	0x00	Intel Management Control 1	

Where:

Intel Management Control 1 is as follows:

Bit #	Default value	Description
0	0b	Enable Critical Session Mode (Keep Phy Link Up and Veto Bit) 0b - Disabled 1b - Enabled When critical session mode is enabled, the following behaviors are disabled: • The PHY is not reset on PE_RST# and PCIe* resets (in-band and link drop). Other reset events are not affected - Internal_Power_On_Reset, device disable, Force TCO, and PHY reset by software. • The PHY does not change its power state. As a result link speed does not change. • The device does not initiate configuration of the PHY to avoid losing link.
1…7	0x0	Reserved

5.4.2.2.10.2 Set Intel Management Control Response (Intel Command 0x20)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..25	0x20	0x00		

5.4.2.2.11 Get Intel Management Control Formats

5.4.2.2.11.1 Get Intel Management Control Command (Intel Command 0x21)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20..21	0x20	0x00		

Where:

Intel Management Control 1 is as described in Section 5.4.2.2.10.2.

5.4.2.2.11.2 Get Intel Management Control Response (Intel Command 0x21)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..26	0x21	0x00	Intel Management Control 1	

5.4.2.2.12 TCO Reset

This command causes the NC to perform TCO reset, if force TCO reset is enabled in the EEPROM.

If the BMC has detected that the OS is hung and has blocked the Rx/Tx path, the force TCO reset clears the data-path (Rx/Tx) of the NC to enable the BMC to transmit/receive packets through the NC.

When this command is issued to a channel in a package, it applies only to the specific channel.

After successfully performing the command, the NC considers the Force TCO command as an indication that the OS is hung and clears the DRV_LOAD flag (disable the LAN device driver).

5.4.2.2.12.1 Perform Intel TCO Reset Command (Intel Command 0x22)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20	0x22			

5.4.2.2.12.2 Perform Intel TCO Reset Response (Intel Command 0x22)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..26	0x22			

5.4.2.2.13 Checksum Offloading

This command enables the checksum offloading filters in the NC.

When enabled, these filters block any packets that did not pass IP, UDP and TCP checksums from being forwarded to the MC.

5.4.2.2.13.1 Enable Checksum Offloading Command (Intel Command 0x23)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20	0x23			

5.4.2.2.13.2 Enable Checksum Offloading Response (Intel Command 0x23)

	Bits			
Bytes	**31..24**	**23..16**	**15..08**	**07..00**
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..26	0x23			

5.4.2.2.13.3 Disable Checksum Offloading Command (Intel Command 0x24)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Manufacturer ID (Intel 0x157)			
20	0x24			

5.4.2.2.13.4 Disable Checksum Offloading Response (Intel Command 0x24)

	Bits			
Bytes	31..24	23..16	15..08	07..00
00..15	NC-SI Header			
16..19	Response Code		Reason Code	
20..23	Manufacturer ID (Intel 0x157)			
24..26	0x24			

5.4.3 Basic NC-SI Workflows

5.4.3.1 Package States

A NC package can be in one of the following two states:

1. Selected - In this state, the package is allowed to use the NC-SI lines, meaning the NC package might send data to the BMC.
2. De-selected - In this state, the package is not allowed to use the NC-SI lines, meaning, the NC package cannot send data to the BMC.

Also note that the MC must select no more than one NC package at any given time.

Package selection can be accomplished in one of two methods:

1. Select Package command - this command explicitly selects the NC package.
2. Any other command targeted to a channel in the package also implicitly selects that NC package.

Package de-select can be accomplished only by issuing the De-Select Package command.

Note: The MC should always issue the Select Package command as the first command to the package before issuing channel-specific commands.

For further details on package selection, refer to the NC-SI specification.

5.4.3.2 Channel States

A NC channel can be in one of the following states:

1. Initial State - In this state, the channel only accepts the Clear Initial State command (the package also accepts the Select Package and De-Select Package commands).
2. Active state - This is the normal operational mode. All commands are accepted.

For normal operation mode, the MC should always send the Clear Initial State command as the first command to the channel.

5.4.3.3 Discovery

After interface power-up, the MC should perform a discovery process to discover the NCs that are connected to it.

This process should include an algorithm similar to the following:

1. For package_id=0x0 to MAX_PACKAGE_ID
   a. Issue Select Package command to package ID package_id
   b. If received a response then

      For internal_channel_id = 0x0 to MAX_INTERNAL_CHANNEL_ID

      Issue a Clear Initial State command for package_id | internal_channel_id (the combination of package_id and internal_channel_id to create the channel ID).

      If a response was received then

      Consider internal_channel_id as a valid channel for the package_id package

      The MC can now optionally discover channel capabilities and version ID for the channel

      Else (If not a response was not received, then issue a Clear Initial State command three times.

      Issue a De-Select Package command to the package (and continue to the next package).
   c. Else (If a response was not received, issue a Select Packet command three times.

5.4.3.4 Configurations

This section details different configurations that should be performed by the MC.

It is considered a good practice that the MC does not consider any configuration valid unless the MC has explicitly configured it after every reset (entry into the initial state).

As a result, it is recommended that the MC re-configure everything at power-up and channel/package resets.

5.4.3.4.1 NC Capabilities Advertisement

NC-SI defines the Get Capabilities command. It is recommended that the MC use this command and verify that the capabilities match its requirements before performing any configurations.

For example, the MC should verify that the NC supports a specific AEN before enabling it.

5.4.3.4.2 Receive Filtering

In order to receive traffic, the MC must configure the NC with receive filtering rules. These rules are checked on every packet received on the LAN interface (such as from the network). Only if the rules matched, will the packet be forwarded to the MC.

5.4.3.4.2.1 MAC Address Filtering

NC-SI defines three types of MAC address filters: unicast, multicast and broadcast. To be received (not dropped) a packet must match at least one of these filters.

Note: The MC should set one MAC address using the Set MAC Address command and enable broadcast and global multicast filtering.

Unicast/Exact Match (Set MAC Address Command)

This filter filters on specific 48-bit MAC addresses. The MC must configure this filter with a dedicated MAC address.

Note: The NC might expose three types of unicast/exact match filters (such as MAC filters that match on the entire 48 bits of the MAC address): unicast, multicast and mixed. The 82598 exposes two mixed filters, which might be used both for unicast and multicast filtering. The MC should use one mixed filter for its MAC address.

Refer to NC-SI specification - Set MAC Address for further details.

Broadcast (Enable/Disable Broadcast Filter Command)

NC-SI defines a broadcast filtering mechanism which has the following states:

1. Enabled - All broadcast traffic is blocked (not forwarded) to the MC, except for specific filters (such as ARP request, DHCP, and NetBIOS).
2. Disabled - All broadcast traffic is forwarded to the MC, with no exceptions.

Note: Refer to NC-SI specification Enable/Disable Broadcast Filter command.

Global Multicast (Enable/Disable Global Multicast Filter)

NC-SI defines a multicast filtering mechanism which has the following states:

1. Enabled - All multicast traffic is blocked (not forwarded) to the MC.
2. Disabled - All multicast traffic is forwarded to the MC, with no exceptions.

The recommended operational mode is Enabled, with specific filters set.

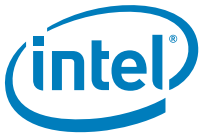
Note: Not all multicast filtering modes are necessarily supported.

Refer to NC-SI specification Enable/Disable Global Multicast Filter command for further details.

5.4.3.4.2.2 VLAN

NC-SI defines the following VLAN work modes:

Mode	Command and Name	Descriptions
Disabled	Disable VLAN command	In this mode, no VLAN frames are received.
Enabled #1	Enable VLAN command with VLAN only	In this mode, only packets that matched a VLAN filter are forwarded to the MC.
Enabled #2	Enable VLAN command with VLAN only + non-VLAN	In this mode, packets from mode 1 + non-VLAN packets are forwarded.
Enabled #3	Enable VLAN command with Any-VLAN + non-VLAN	In this mode, packets are forwarded regardless of their VLAN state.



Refer to NC-SI specification - Enable VLAN command for further details.

The 82598 only supports modes #1 and #3.

Recommendation:

1. Modes:
   a. If VLAN is not required - use the disabled mode.
   b. If VLAN is required - use the enabled #1 mode.
2. If enabling VLAN, The MC should also set the active VLAN ID filters using the NC-SI Set VLAN Filter command prior to setting the VLAN mode.

5.4.3.5 Pass-Through Traffic States

The MC has independent, separate controls for enablement states of the receive (from LAN) and of the transmit (to LAN) pass-through paths.

5.4.3.5.1 Channel Enable

This mode controls the state of the receive path:

1. Disabled: The channel does not pass any traffic from the network to the MC.
2. Enabled: The channel passes any traffic from the network (that matched the configured filters) to the MC.

Note: This state also affects AENs: AENs is only sent in the enabled state.

Note: The default state is disabled.

Note: It is recommended that the MC complete all filtering configuration before enabling the channel.

5.4.3.5.2 Network Transmit Enable

This mode controls the state of the transmit path:

1. Disabled - the channel does not pass any traffic from the MC to the network.
2. Enabled - the channel passes any traffic from the MC (that matched the source MAC address filters) to the network.

Note: The default state is disabled.

Note: The NC filters pass-through packets according to their source MAC address. The NC tries to match that source MAC address to one of the MAC addresses configured by the Set MAC Address command. As a result, the MC should enable network transmit only after configuring the MAC address.

Note: It is recommended that the MC complete all filtering configuration (especially MAC addresses) before enabling the network transmit.

Note: This feature can be used for fail-over scenarios. See Section 5.4.5.3.

5.4.3.6 Asynchronous Event Notifications

The asynchronous event notifications are unsolicited messages sent from the NC to the MC to report status changes (such as link change, OS state change, etc.).

Recommendations:

- The MC firmware designer should use AENs. To do so, the designer must take into account the possibility that a NC-SI response frame (such as a frame with the NC-SI EtherType), arrives out-of-context (not immediately after a command, but rather after an out-of-context AEN).
- To enable AENs, the MC should first query which AENs are supported, using the Get Capabilities command, then enable desired AEN(s) using the Enable AEN command, and only then enable the channel using the Enable Channel command.

5.4.3.7 Querying Active Parameters

The MC can use the Get Parameters command to query the current status of the operational parameters.

5.4.4 Resets

In NC-SI there are two types of resets defined:

1. Synchronous entry into the initial state.
2. Asynchronous entry into the initial state.

Recommendations:

- It is very important that the MC firmware designer keep in mind that following any type of reset, all configurations are considered as lost and thus the MC must re-configure everything.
- As an asynchronous entry into the initial state might not be reported and/or explicitly noticed, the MC should periodically poll the NC with NC-SI commands (such as Get Version ID, Get Parameters, etc.) to verify that the channel is not in the initial state. Should the NC channel respond to the command with a Clear Initial State Command Expected reason code - The MC should consider the channel (and most probably the entire NC package) as if it underwent a (possibly unexpected) reset event. Thus, the MC should re-configure the NC. See the NC-SI specification section on Detecting Pass-through Traffic Interruption.
- The Intel recommended polling interval is 2-3 seconds.

For exact details on the resets, refer to NC-SI specification.

5.4.5 Advanced Workflows

5.4.5.1 Multi-NC Arbitration

As described in Section 5.4.1.2, in a multi-NC environment, there is a need to arbitrate the NC-SI lines.

Figure 5-15 shows the system topology of such an environment.

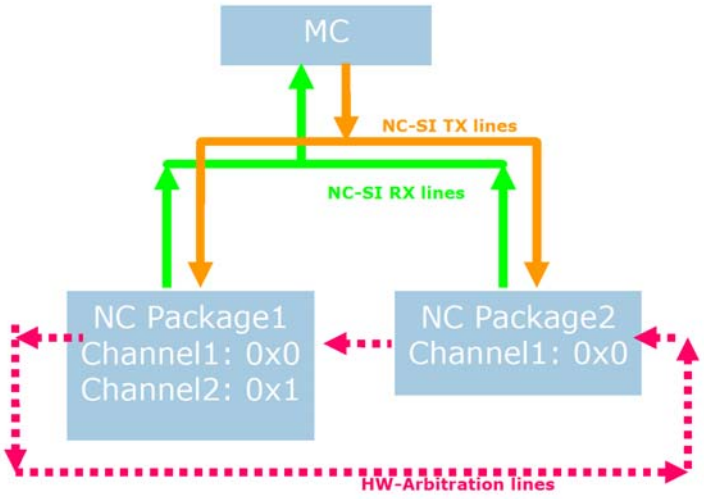


Figure 5-15. Multi-NC Environment

See Figure 5-15. The NC-SI Rx lines are shared between the NCs. To enable sharing of the NC-SI Rx lines, NC-SI has defined an arbitration scheme.

The arbitration scheme mandates that only one NC package can use the NC-SI Rx lines at any given time. The NC package that is allowed to use these lines is defined as selected. All the other NC packages are de-selected.

NC-SI has defined two mechanisms for the arbitration scheme:

1. Package Selection by the MC. In this mechanism, the MC is responsible for arbitrating between the packages by issuing NC-SI commands (Select/De-Select Package). The MC is responsible for having only one package selected at any given time.
2. HW Arbitration. In this mechanism, two additional pins on each NC package are used to synchronize the NC package. Each NC package has an ARB_IN and ARB_OUT line and these lines are used to transfer Tokens. A NC package that has a token is considered selected.

Note: Hardware arbitration is enabled by default after interface power-up.

Note: The 82598 does not support hardware arbitration.

For further details, refer to section 4 in the NC-SI specification.

5.4.5.1.1 Package Selection Sequence Example

Following is an example work flow for a MC and occurs after the discovery, initialization, and configuration.

Assuming the MC needs to share the NC-SI bus between packages the MC should:

1. Define a time-slot for each device.
2. Discover, initialize, and configure all the NC packages and channels.
3. Issue a De-Select Package command to all the channels.
4. Set active_package to 0x0 (or the lowest existing package ID).

5. At the beginning of each time slot the MC should:
   a. Issue a De-Select Package to the active_package. The MC must then wait for a response and then an additional timeout for the package to become de-selected (200 µs). See the NC-SI specification table 10 - parameter NC Deselect to Hi-Z Interval.
   b. Find the next available package (typically active_package = active_package + 1).
   c. Issue a Select Package command to active_package.

5.4.5.2 External Link Control

The MC can use the NC-SI Set Link command to control the external interface link settings. This command enables the MC to set the auto-negotiation, link speed, duplex, and other parameters.

This command is only available when the host OS is not present. Indicating the host OS status can be obtained via the Get Link Status command and/or Host OS Status Change AEN command.

Recommendation:

- Unless explicitly needed, it is not recommended to use this feature. The NC-SI Set Link command does not expose all the possible link settings and/or features. This might cause issues under different scenarios. Even if decided to use this feature, it is recommended to use it only if the link is down (trust the 82598 until proven otherwise).
- It is recommended that the MC first query the link status using the Get Link Status command. The MC should then use this data as a basis and change only the needed parameters when issuing the Set Link command.

For further details, refer to the NC-SI specification.

5.4.5.2.1 Set Link While LAN PCIe Functionality is Disabled

In cases where a LAN device is used solely for manageability and its LAN PCIe function is disabled, using the NC-SI Set Link command while advertising multiple speeds and enabling auto-negotiation results in the lowest possible speed chosen.

To enable link of higher a speed, the MC should not advertise speeds that are below the desired link speed, as the lowest advertised link speed is chosen.

When the LAN device is only used for manageability and the link speed advertisement is configured by the MC, changes in the power state of the LAN device is not effected and the link speed is not re-negotiated by the LAN device.

5.4.5.3 Multiple Channels (Fail-Over)

In order to support a fail-over scenario, it is required from the MC to operate two or more channels. These channels might or might not be in the same package.

The key element of a fault-tolerance fail-over scenario is having two (or more) channels identifying to the switch with the same MAC address, but only one of them being active at any given time (such as switching the MAC address between channels).

To accomplish this, NC-SI provides the following:

1. Enable Network Tx command. This command enables shutting off the network transmit path of a specific channel. This enables the MC to configure all the participating channels with the same MAC address but only enable one of them.
2. Link Status Change AEN or Get Link Status command.

5.4.5.3.1 Fail-Over Algorithm Example

Following is a sample workflow for a fail-over scenario for the 82598 dual-port GbE controller (one package and two channels).

1. MC initializes and configures both channels after power-up. However, the MC uses the same MAC address for both of the channels.
2. The MC queries the link status of all the participating channels. The MC should continuously monitor the link status of these channels. This can be accomplished by listening to AENs (if used) and/or periodically polling using the Get Link Status command.
3. The MC then only enables channel 0 for network transmission.
4. The MC then issues a gratuitous ARP (or any other packet with its source MAC address) to the network. This packet informs the switch that this specific MAC address is registered to channel 0's specific LAN port.
5. The MC begins normal workflow.
6. Should the MC receive an indication (AEN or polling) that the link status for the active channel (channel 0) has changed, the MC should:
   a. Disable channel0 for Network Transmission.
   b. Check if a different channel is available (link is up).

      If found:

      Enable network Tx for that specific channel.

      Issue a gratuitous ARP (or any other packet with its source MAC address) to the network. This packet informs the switch that this specific MAC address is registered to channel 0's specific LAN port.

      Resume normal workflow.

      If not found, report the error and continue polling until a valid channel is found.

Note: The above algorithm can be generalized such that the start-up and normal workflow are the same.

In addition, the MC might need to use a specific channel (such as channel 0). In this case, the MC should switch the network transmit to that specific channel as soon as that channel becomes valid (link is up).

Recommendations:

- It is recommended to wait a link-down-tolerance timeout before a channel is considered invalid. For example, a link re-negotiation might take a few seconds (normally 2 to 3 or might be up to 9). Thus, the link must be re-established after a short time.
- Typically, this timeout is recommended to be three seconds.
- Even when enabling and using AENs, it is still recommended to periodically poll the link status, as dropped AENs might not be detected.

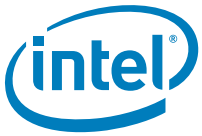
5.4.5.4 Statistics

The MC might use the statistics commands as defined in NC-SI. These counters are meant mostly for debug purposes and are not all supported.

The statistics are divided into three commands:

1. Controller statistics - These are statistics on the primary interface (to the host OS). See the NC-SI specification for details.
2. NC-SI statistics - These are statistics on the NC-SI control frames (such as commands, responses, AENs, etc.). See the NC-SI specification for details.
3. NC-SI pass-through statistics - These are statistics on the NC-SI pass-through frames. See the NC-SI specification for details.

§ §



6 Mechanical Specification

This section describes the 82598 physical characteristics.

6.1 Package Information

The controller is a 883-lead flip-chip ball grid array (FC-BGA) measuring 31 mm by 31 mm. The nominal ball pitch is 1 mm.

Note: Empty spots in the following mechanical pin diagram are correct.

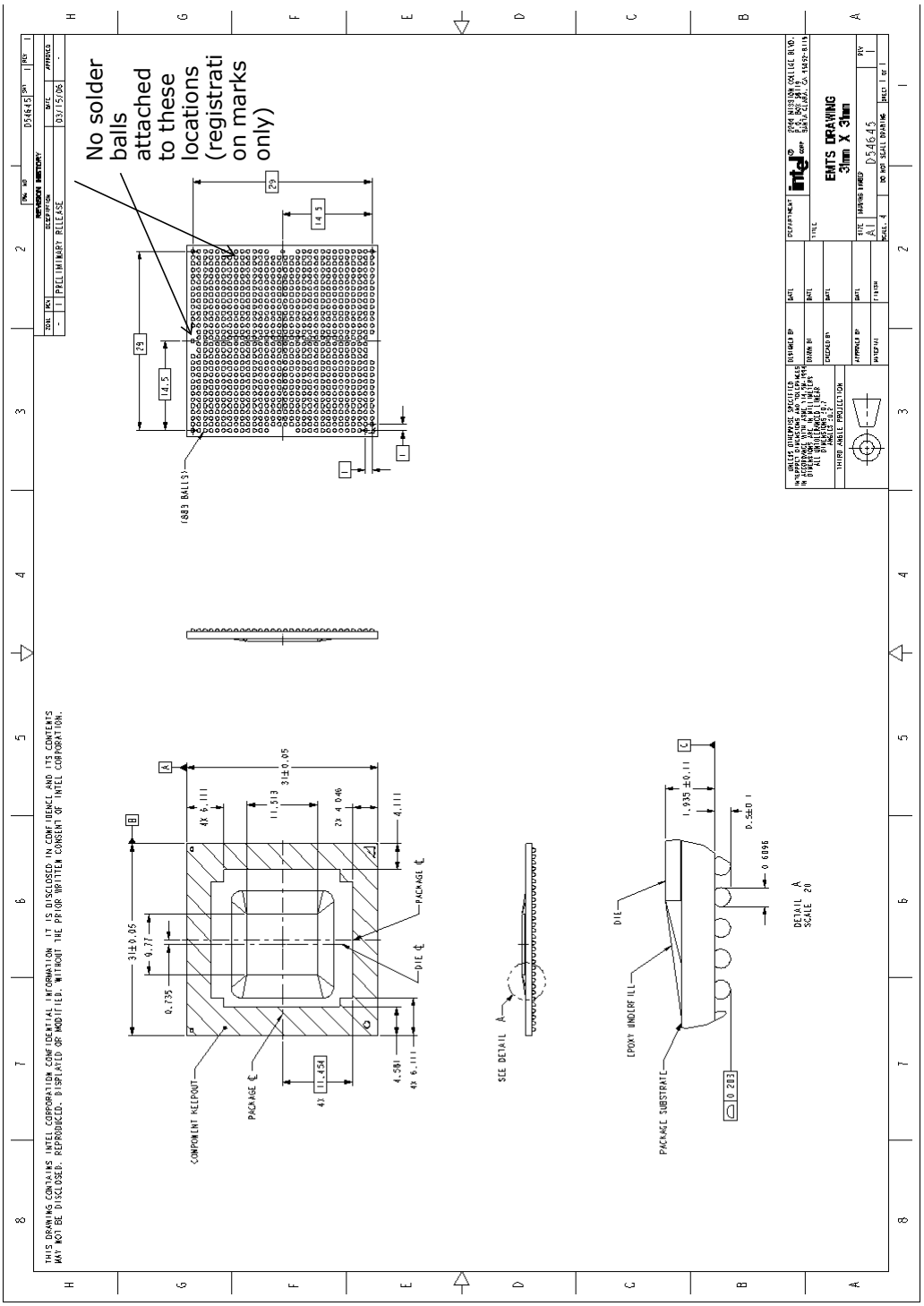


Figure 6-1. Mechanical Specifications

§ §

7 Electrical Specifications

7.1 Operating Conditions

This chapter describes 82598 DC and AC (timing) electrical characteristics. This includes maximum rating, recommended operating conditions, power sequencing requirements, and DC and AC timing specifications. DC and AC characteristics include generic digital 3.3 V dc IO specification, as well as other specifications supported by the device.

7.2 Absolute Maximum Ratings

Table 7-1. Absolute Maximum Ratings

Symbol	Parameter	Min	Max	Units
T_{case}	Case Temperature Under Bias	0	105	°C
$T_{storage}$	Storage Temperature Range	-65	140	°C
Vi/Vo	3.3 V dc Compatible I/Os Voltage Analog 1.2 I/O Voltage Analog 1.8 I/O Voltage	Vss-0.5 Vss–0.2 Vss-0.3	4.60 1.68 2.52	V dc
VCC3P3	3.3 V dc Periphery DC Supply Voltage	Vss -0.5	4.60	V dc
VCC1P8	1.8 V dc Analog DC Supply Voltage	Vss-0.3	2.52	V dc
VCC1P2	1.2 V dc Core/Analog DC Supply Voltage	Vss-0.2	1.68	V dc

Note: Ratings in this table 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 may affect device reliability.

Recommended operation conditions require accuracy of power supply of +/-5% relative to the nominal voltage

Vi/Vo of 3.3 V dc compatible I/Os maximum should be the minimum of 4.6 V dc or (VCC3P3+0.5).

7.3 Recommended Operating Conditions

Table 7-2. Recommended Operating Conditions

Symbol	Parameter	Min	Max	Units
Ta	Operating Temperature Range Commercial (Ambient, 0 CFS airflow)	0	55	°C

Note: For normal device operation, adhere to the limits in this table. Sustained operations of a device at conditions exceeding these values, even if 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.

Recommended operation conditions require a power supply accuracy of +/-5%, relative to the nominal voltage.

External Heat Sink (EHS) needed.

7.4 Power Delivery

7.4.1 Power Supply Specifications

Table 7-3. 3.3 V dc Supply Voltage Requirements

Parameters	Description	Min	Max	Units
Rise Time	Time from 10% to 90% mark	0.1	100	ms
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*V(min)/Rise time (max) Max: 0.8*V(max)/Rise time (min)	24	28800	V/s
Operational Range	Voltage range for normal operating conditions	3	3.6	V dc
Ripple	Maximum voltage ripple (peak to peak)	N/A	70	mV
Overshoot	Maximum overshoot allowed	N/A	100	mV
Overshoot Settling Time	Maximum overshoot allowed duration. (At that time delta voltage should be lower than 5mv from steady state voltage)	N/A	0.05	ms
Suggested Decoupling Capacitance	Capacitance Range	20	47	µF
Capacitance ESR	Equivalent series resistance of output capacitance	N/A	50	MΩ

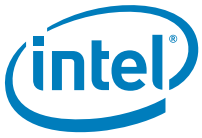


Table 7-4. 1.8 V dc Supply Voltage Requirements

Parameters	Description	Min	Max	Units
Rise Time	Time from 10% to 90% mark	0.1	100	ms
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*V(min)/Rise time (max) Max: 0.8*V(max)/Rise time (min)	14	15000	V/s
Operational Range	Voltage range for normal operating conditions	1.71	1.89	V dc
Ripple	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
Suggested Decoupling Capacitance	Capacitance range	50	75	µF
Capacitance ESR	Equivalent series resistance of output capacitance	N/A	50	MΩ

Table 7-5. 1.2 V dc Supply Voltage Requirements

Parameter	Description	Min	Max	Units
Rise Time	Time from 10% to 90% mark	0.1	100	ms
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*V(min)/Rise time (max) Max: 0.8*V(max)/Rise time (min)	9.1	10000	V/s
Operational Range	Voltage range for normal operating conditions	1.14	1.26	V dc
Ripple	Maximum voltage ripple (peak to peak)	N/A	40	mV
Overshoot	Maximum overshoot allowed	N/A	100	mV
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
Suggested Decoupling Capacitance	Capacitance range	150	175	µF
Capacitance ESR	Equivalent series resistance of output capacitance	5	50	MΩ

7.4.2 Power Supply Sequencing

The following relationships between rise times of different power supplies should be maintained to avoid risk of either latch-up or forward-biased internal diodes:

($T_{3.3\ V\ dc\ supply} \leq T_{1.8\ V\ dc\ supply}$) AND ($T_{3.3\ V\ dc\ supply} \leq T_{1.2\ V\ dc\ supply}$)

(1.8 V dc supply $\leq$ 3.3 V dc supply) AND (1.2 V dc supply $\leq$ 3.3 V dc supply)

On power-on, after 3.3 V dc reaches 10% of its final value, voltage rails of 1.8 V dc and 1.2 V dc are allowed 100 ms (T_{3_18}) to reach final operating values. To keep current leakage at a minimum, turn the rails on almost simultaneously. See Figure 7-1 for the relationship requirements (between 1.8 V dc and 1.2 V dc power supplies).

For the fastest 1.2 V dc ramp: the $V_{1.2_1.8}$ parameter value (maximum voltage difference) should be less than 0.3 V dc. For the slowest 1.2 V dc ramp: the $T_{1.2_1.8}$ parameter value (maximum time difference), for below the 0.4 V dc level, should be less than 500 µs.

For power-down, turn off all rails at the same time and leave voltage to decay.

Table 7-6. Power Sequencing

Symbol	Parameter	Min	Max	units
$T_{3_1.8}$	VCC3P3 (3.3 V dc) stable to VCC1P8 (1.8 V dc) stable	0.01	100	ms
$V_{1.2_1.8}$	VCC1P2 (1.2 V dc) to VCC1P8 (1.8 V dc) ramp difference - fastest[1]	0	0.3	V
$T_{1.8_1.2}$	VCC1P8 (1.8 V dc) to VCC1P2 (1.2 V dc) - slowest[1,2]	N/A	0.5	ms
$T_{3_1.2}$	VCC3P3 (3.3 V dc) stable to VCC1P2 (1.2 V dc) stable	0.01	100.5	ms

1. If board designers have difficulty meeting this parameter, use the LAN_PWR_GOOD (external) timing parameter instead (see Section 7.6.6.1).
2. Measured at a level of 0.4 V dc.

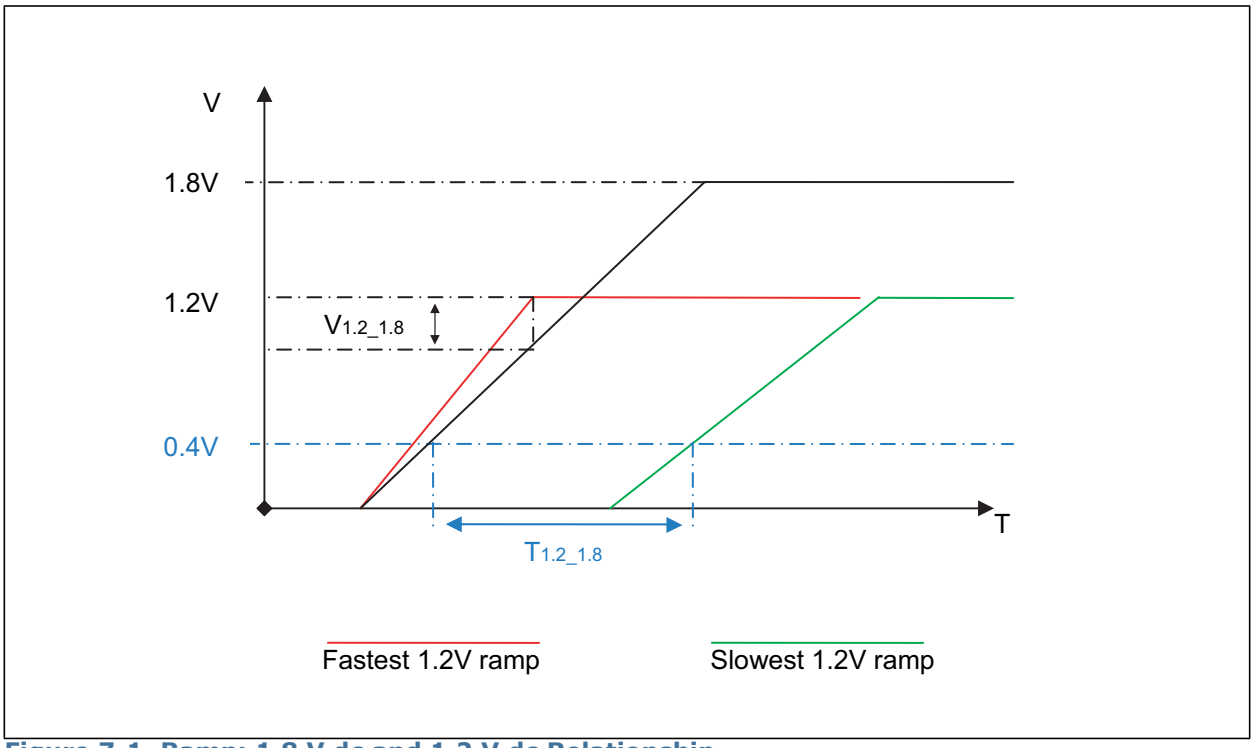


Figure 7-1. Ramp: 1.8 V dc and 1.2 V dc Relationship

Figure notes:

- If VCC1P2 (1.2 V dc) leading (fastest) VCC1P8 (1.8 VDC), VCC1P2 (1.2 V dc) must not be more than 0.3 V dc maximum voltage difference.
- If VCC1P2 (1.2 V dc) lagging (slowest) VCC1P8 (1.8 VDC), VCC1P2 (1.2 V dc) must not lag VCC1P8 (1.8 V dc) by more than 0.5 ms maximum (measured at 0.4 V dc).

7.4.3 Power Consumption

The following tables show targets for device power. The numbers below apply to device current and power and do not include power losses on external components. Other than TDP, assume typical temperature, silicon, and Vcc.

Table 7-7. Power Consumption – Dual-Port – D0 – Active Link (Dual Port Functionality)

	@1000 Mb/s	@ 10 Gb/s	
	Typ Icc (mA)	Typ Icc (mA)	Max Icc (mA)
3.3 V dc	21	21	21
1.8 V dc	289	289	289
1.2 V dc	1903	3502	4925
Total Device Power (mW)	2873	4791	6500

Note: a. Typical conditions: room temperature (TA) = 25 °C, nominal voltages and continuous network traffic at link speed.

Table 7-8. Power Consumption – D3 State (Dual Port Functionality)

	WoL Enabled @ 1000 Mb/s	Device Disabled[c]
	Typ Icc (mA)	Typ Icc (mA)
3.3 V dc	21	21
1.8 V dc	51	51
1.2 V dc	1244	486
Total Device Power (mW)	1654	744

Note:

Table 7-9. Power Consumption – Single Port – D0 – Active Link/Second Port Disabled

	@ 1000 Mb/s	@ 10 Gb/s	
	Typ Icc (mA)	Typ Icc (mA)	Max Icc (mA)
3.3 V dc	21	21	21
1.8 V dc	289	289	289
1.2 V dc	1484	2378	4016
Total Device Power (mW)	2370	3442	5409

Note:

7.5 DC Specifications

7.5.1 Digital I/O

Table 7-10. Digital I/O DC Specification

Symbol	Parameter	Conditions	Min	Max	Units	Note
VOH	Output High Voltage	IOH = -16mA; VCC3P3 = Min	2.4		V dc	
VOL	Output Low Voltage	IOL = 10mA; VCC=Min		0.4	V dc	
VOLled	LED Output Low Voltage	IOL = 12mA; VCC3P3 = Min		0.4	V dc	
VIH	Input High Voltage		2.0	VCC3P3 + 0.3	V dc	1
VIL	Input Low Voltage		-0.3	0.8	V dc	1
Iil	Input Current	VCC3P3 = Max; VI =3.6V/GND		15	µA	
IOFF	Current at IDDQ Mode			50	µA	3
PU	Internal Pull Up		3	8	KΩ	2,3,4, 5
Ipup	Internal Pull Up Current	0-0.5*VCC3P3[V]	0.43	1	mA	6, 7
	Built-in Hysteresis		100	400	mV	
Cin	Pin Capacitance	Not measured, only characterized	3	7	pF	
Cload	Load Capacitance	Timing characterized with this load	0	16	pF	

Note:

1. The input buffer also has hysteresis > 80 mV.
2. Internal pull-up maximum was characterized at slow corner (110 °C, VCC3P3=min, process slow)
3. Internal pull-up minimum was characterized at fast corner (0 °C, VCC3P3=max, process fast).
4. External R pull-down recommended ≤ 400 Ω.
5. External R pull-up recommended ≤ 3 KΩ.
6. External buffer recommended strength ≥ 2 mA.
7. Internal pull-up maximum current consumption was characterized at fast corner (0 °C, VCC3P3=max, process fast)
   Internal pull-up minimum current consumption was characterized at slow corner (110 °C, VCC3P3=min, process slow).

   The previous table applies to PE_RST_N, LED0[3:0], LED1[3:0], POR_BYPASS, Internal Power On Reset, LAN_PWR_GOOD, MAIN_PWR_OK, JTCK, JTDI, JTDO, JTMS, JRST_N, SDP0[3:0], SDP1[3:0], FLSH_SI, FLSH_SO, FLSH_SCK, FLSH_CE_N, EE_DI, EE_DO, EE_SK, EE_CS_N.

7.5.2 Open Drain I/O

Table 7-11. Open Drain DC Specification

Symbol	Parameter	Condition	Min	Max	Units	Note
V_{ih}	Input High Voltage		2.1		V dc	
V_{il}	Input Low Voltage			0.8	V dc	
$I_{leakage}$	Output Leakage Current	0 < Vin < VCC3P3		+/-10	µA	2
V_{ol}	Output Low Voltage	@ Ipullup		0.4	V	5
I_{pullup}	Current sinking	Vol=0.4V	4		mA	
C_{in}	Input Pin Capacitance			7	pF	3
C_{load}	Max load Pin Capacitance			30	pF	3
I_{offsmb}	Input leakage current	VCC3P3 off or floating		+/-10	µA	2

Note:
1. Table applies to SMBD, SMBCLK, SMBALRT _N, PE_WAKE_N.
2. Device meets this, powered or not.
3. Characterized, not tested.
4. Cload should be calculated according to the external pull-up resistor and the frequency.
5. OD no high output drive. VOL max=0.4 V dc at 16 mA, VOL max=0.2 V dc at 0.1 mA.
6.

7.5.3 NC-SI I/O

Table 7-12. NC-SI Input and Output Pads DC Specification

Symbol	Parameter	Conditions	Min	Max	Units
VOH	Output High Voltage	IOH = -4 mA; VCC3P3 = Min	2.4		V dc
VOL	Output Low Voltage	IOL = 4mA; VCC3P3 = Min		0.4	V dc
VIH	Input High Voltage		2.0		V dc
VIL	Input Low Voltage			0.8	V dc
Vihyst	Input Hysteresis		100		mV
Iil/Iih	Input Current	VCC3P3 = Max; Vin =3.6V/GND		15	µA
Cin	Input Capacitance			5	pF
Ipup	Pull-Up Current	Vout = 0V (GND)	0.4	1.3	mA

7.6 Digital I/F AC Specifications

7.6.1 Digital I/O AC Specification

Table 7-13. Digital I/O AC Specification

Parameters	Description	Min	Max	Cload	Note
Tor	Output Time rise	0.2 ns	1 ns	16 pF	
Tof	Output Time fall	0.2 ns	1 ns		
Todr	Output delay rise	0.8 ns	3 ns		
Todf	Output delay fall	0.8 ns	3 ns		

The input delay test conditions: Maximum input level = VIN = 2.7V; Input rise/fall time (0.2VIN to 0.8VIN) = 1ns (Slew Rate ~ 1.5ns).

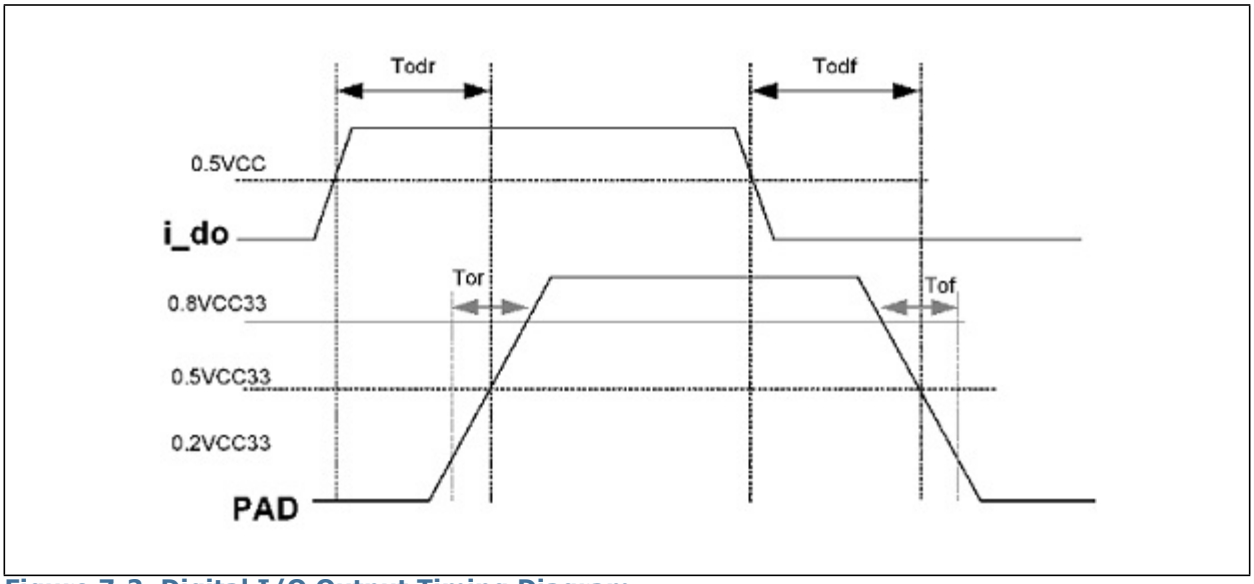


Figure 7-2. Digital I/O Output Timing Diagram

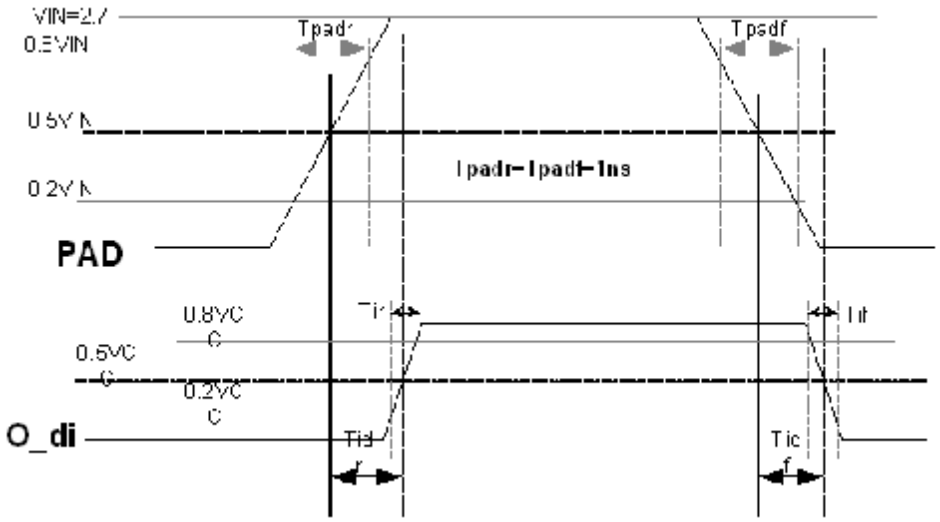


Figure 7-3. Digital I/O Input Timing Diagram

7.6.2 EEPROM AC Specifications

Information in this table is applicable over recommended operating range from Ta = 0 °C to +85 °C, VCC3P3 = 3.3 V dc, Cload = 1 TTL Gate and 16 pF (unless otherwise noted).

Table 7-14. EEPROM AC Timing Specifications

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{SCK}	EE_CK clock frequency	0	2	2.1	MHz	1
t_{RI}	EE_DO rise time		2.5ns	2	µs	
t_{FI}	EE_DO fall time		2.5ns	2	µs	
t_{WH}	EE_CK high time	200	250		ns	2
t_{WL}	EE_CK low time	200	250		ns	
t_{CS}	EE_CS_N high time	250			ns	
t_{CSS}	EE_CS_N setup time	250			ns	
t_{CSH}	EE_CS_N hold time	250			ns	
t_{SU}	Data-in setup time	50			ns	
t_{H}	Data-in hold time	50			ns	
t_{V}	Output valid	0		200	ns	
t_{HO}	Output hold time	0			ns	
t_{DIS}	Output disable time			250	ns	
t_{WC}	Write cycle time			10	ms	

Note: 1. Clock is 2 MHz.
2. 50% duty cycle.

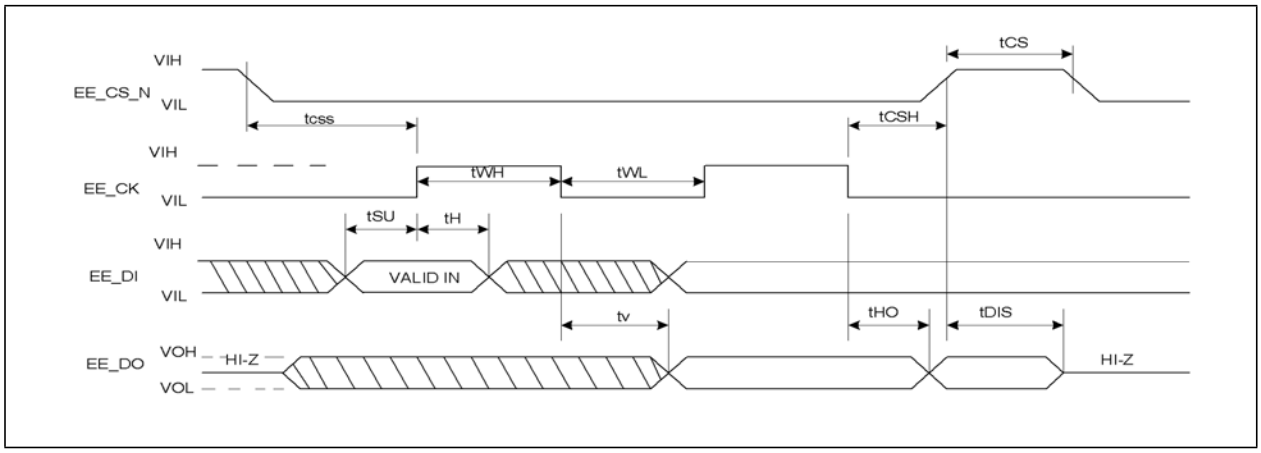
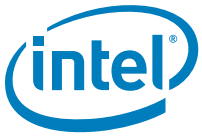


Figure 7-4. EEPROM Timing Characteristics



7.6.3 Flash AC Specification

Information in this table is applicable over recommended operating range from Ta = 0 °C to +85 °C, VCC3P3 = 3.3 V dc, Cload = 1 TTL Gate and 16 pF (unless otherwise noted).

Table 7-15. Flash AC Timing Specification

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{SCK}	FLSH_SCK clock frequency			20	MHz	2
t_{RI}	FLSH_SO rise time		2.5	20	ns	
t_{FI}	FLSH_SO fall time		2.5	20	ns	
t_{WH}	FLSH_SCK high time	20			ns	1
t_{WL}	FLSH_SCK low time	20			ns	1
t_{CS}	FLSH_CE_N high time	25			ns	
t_{CSS}	FLSH_CE_N setup time	25			ns	
t_{CSH}	FLSH_CE_N hold time	25			ns	
t_{SU}	Data-in setup time	5			ns	
t_{H}	Data-in hold time	5			ns	
t_{V}	Output valid			20	ns	
t_{HO}	Output hold time	0			ns	
t_{DIS}	Output disable time			100	ns	
t_{EC}	Erase cycle time per sector			1.1	s	
t_{BPC}	Byte program cycle time		60	100	µs	

Note:
1. 50% duty cycle.
2. Clock is 39.0625 MHz divided by 2.

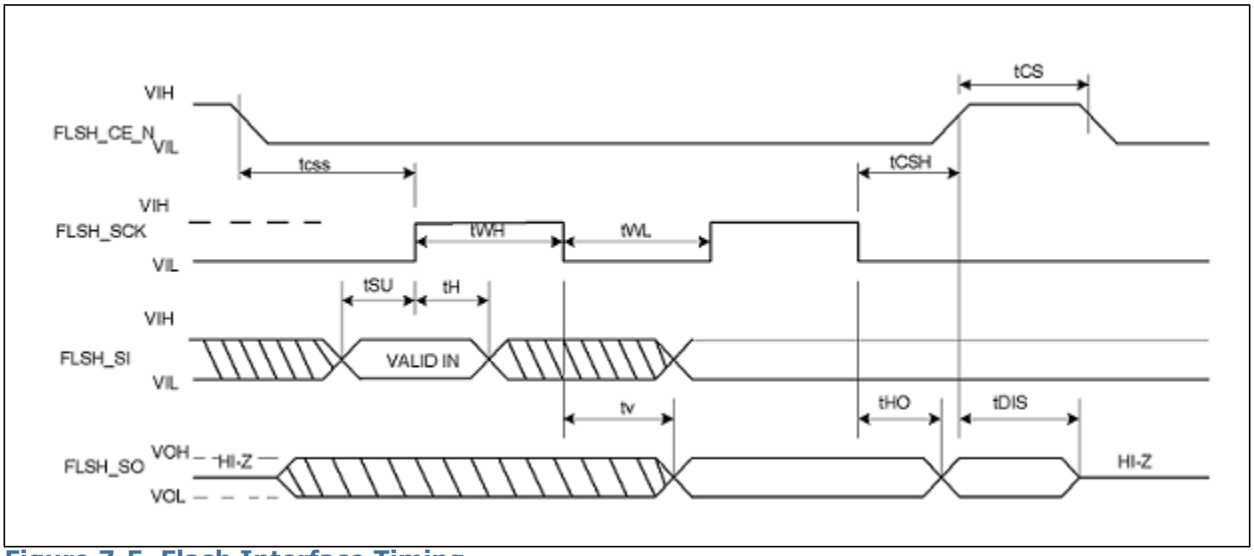


Figure 7-5. Flash Interface Timing

7.6.4 SMBus AC Specification

The 82598 meets the SMBus AC specifications as defined in the SMBus Specification version 2, section 3.1.1. Go to www.smbus.org/specs/ for more details.

The 82598 also supports 400 KHz SMBus (as a slave) and in this case meets the following table.

Table 7-16. SMBus Timing Parameters (Slave Mode)

Symbol	Parameter	Min	Typ	Max	Units
F_{SMB}	SMBus Frequency	10		400	kHz
T_{BUF}	Time between STOP and START	1.44[1]			µs
$T_{HD:STA}$	Hold time after Start Condition. After this period, the first clock is generated.	0.48[1]			µs
$T_{SU:STA}$	Start Condition setup time	1.6[1]			µs
$T_{SU:STO}$	Stop Condition setup time	1.76[1]			µs
$T_{HD:DAT}$	Data hold time	0.32			µs
T_{LOW}	SMBCLK low time	0.8[1]			µs
T_{HIGH}	SMBCLK high time	1.44[1]			µs

1. The actual minimum requirement has to be less. Many of these are below the minimums specified by the SMBus specification.

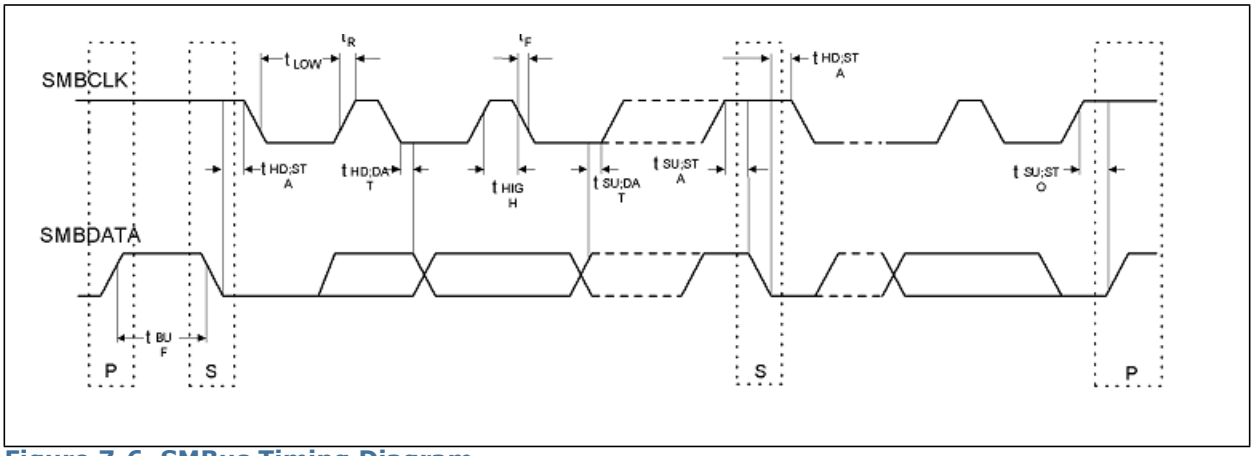
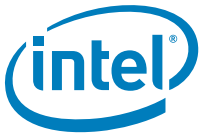


Figure 7-6. SMBus Timing Diagram



7.6.5 NC-SI AC Specification

Table 7-17. NC-SI AC Specification

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Units
REF_CLK Frequency				50	50+100 ppm	MHz
REF_CLK Duty Cycle			35		65	%
Clock-to-out[1]	Tco		2.5		9	ns
Signal Rise Time	Tr	Cload ≤ 25 pF	1		5	ns
		Cload ≤ 50 pF	1		7	ns
Signal Fall Time	Tf	Cload ≤ 25 pF	1		5	ns
		Cload ≤ 50 pF	1		7	ns
Clock Rise Time 1	$Tckr_1$	Cload ≤ 50 pF	0.5		3.5	ns
Clock Rise Time 2	$Tckr_2$		0.5		3.5	ns
Clock Fall Time 1	$Tckf_1$		0.5		3.5	ns
Clock Fall Time 2	$Tckf_2$		0.5		3.5	ns
TXD[1:0], TX_EN, RXD[1:0], CRS_DV, RX_ER Data Setup to REF_CLK rising edge	Tsu		4			ns
TXD[1:0], TX_EN, RXD[1:0], CRS_DV, RX_ER data hold from REF_CLK rising edge	Thold		2			ns
Interface power-up High Impedance Interval	Tpwrz		2			uS
Power Up transient interval (recommendation)	Tpwrt				100	ns
Power Up transient level (recommendation)	Vpwrt		-200		200	mV
Interface power-up Output Enable Interval	Tpwre				10	ms
EXT_CLK Startup Interval	Tclkstrt				100	ms

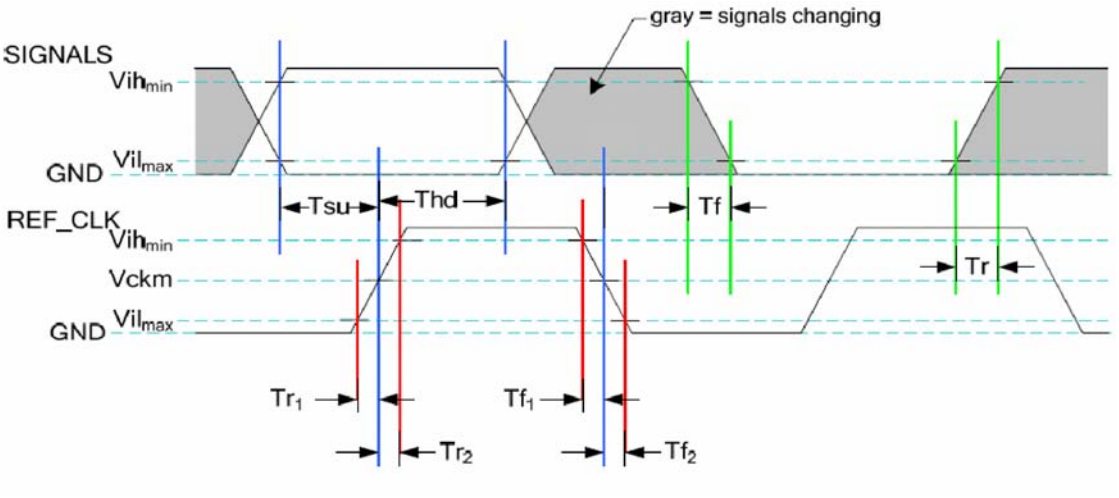


Figure 7-7. NC-SI AC Specifications

7.6.6 Reset Signals

For a power-on indication, the 82598 can either use an Internal Power On Reset indication, which monitors the 1.2 V dc power supply, or an external reset through the LAN_PWR_GOOD pad. The POR_BYPASS pad defines the reset source.

Note: When high, the 82598 uses the LAN_PWR_GOOD pad as a power-on indication. When low, the 82598 uses the Internal Power On Reset circuit.

The timing between the power-up sequence and the different reset signals when using the Internal Power On Reset indication is described in Section 3.2.1.

The POR_BYPASS mode is described in Section 7.6.6.1.

The device power on logic is described in Figure 7-8.

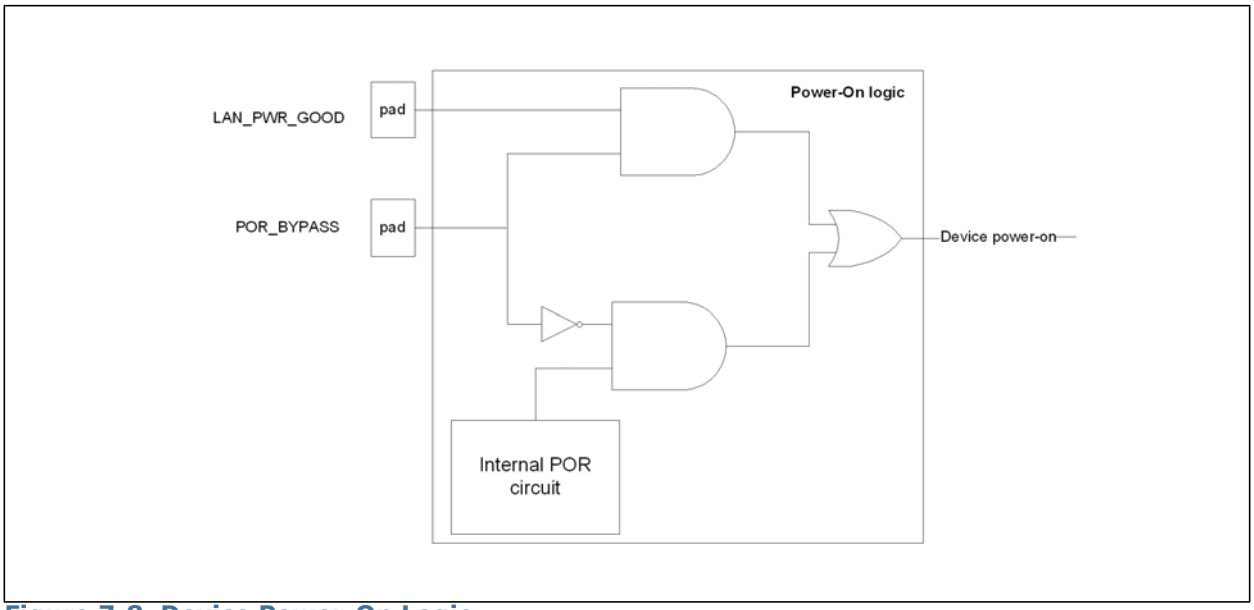


Figure 7-8. Device Power-On Logic

7.6.6.1 POR_BYPASS (External)

When asserting the POR_BYPASS pad, the 82598 uses the LAN_PWR_GOOD pad as a power-on indication that disables the Internal Power On detection circuit. Table 7-18 lists the timing for the External Power On signal.

Table 7-18. Timing for External Power On Signal

Symbol	Title	Description	Min	Max	Units
Tlpgw	LAN_PWR_GOOD minimum width	Minimum width for LAN_PWR_GOOD	10	N/A	µs
Tlpg	LAN_PWR_GOOD low hold	How long it must be low after voltages are in operating range	40	80	ms

LAN_PWR_GOOD and POR_BYPPAS are regular digital I/O signals and their characteristics are described in Section 7.5.1.

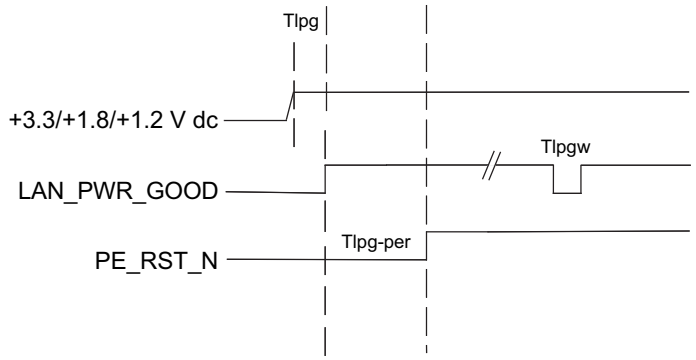


Figure 7-9. LAN_PWR_GOOD Timing

7.6.7 PCIe DC/AC Specification

The transmitter and receiver specifications are available in the PCIe Card Electromechanical Specification revision 1.1.

7.6.7.1 PCIe Specification (Receiver and Transmitter)

Refer to the PCIe specification.

7.6.7.2 PCIe Specification (Input Clock)

The input clock for PCIe relates to a differential input clock in a frequency of 100 MHz. For more details, refer to the PCIe Card Electromechanical specifications (refclk specifications).

7.6.8 Reference Clock Specification

The external clock must be 156.25 MHz +/-0.005% (+/- 50 ppm). Refer to Table 7-19. VDD in the table refers to the 1.2 V dc supply.

Table 7-19. Input Reference Clock DC Specification Requirements

Parameter	Minimum	Typical	Maximum	Unit
RCLKEXTP/N (Differential Input Voltage)	1000		2000	mV (p-p)
RCLKEXTP/N (Input Common Mode Voltage Range)	0.30 (VDD/2-300 mV)	0.60 (VDD/2)	0.90 (VDD/2+300 mV)	V dc
RCLKEXTP/N (Input Bias Voltage)[1]	0.67 (VDD*0.64-100 mV)	0.77 (VDD*0.64)	0.87 (VDD*0.64+100 mV)	V dc
RCLKEXTP/N (Differential Input Jitter)[2]			3	pS/RMS
RCLKEXTP/N (Differential Input Resistance)		10K		Ω
RCLKEXTP/N (Single Ended Capacitance)			5	pF

1. This is the voltage that both RCLKEXTP and RCLKEXTN are internally biased to.
2. 10 Hz to 20 Hz.

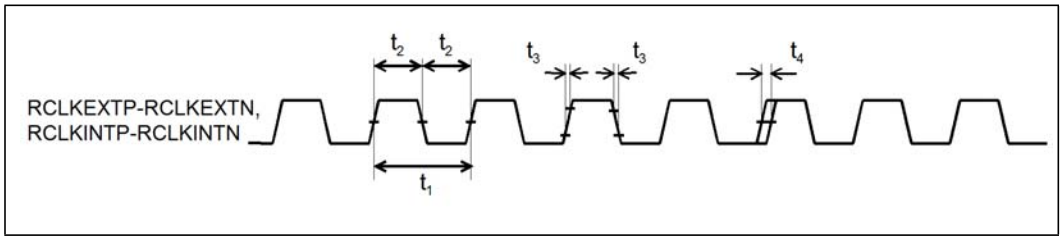


Figure 7-10. Reference Clock AC Characteristics

Table 7-20. Reference Clock AC Characteristics

Symbol	Value	Name
t1	6.4 nS (typical)	Input Clock Frequency
t2	45%-55%	Input Duty Cycle
t3	500 ps-1 ns	Input Rise and Fall Time
t4	3.0 pS, RMS (maximum)	Differential Input Jitter

§ §



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

8 Design Guidelines

This section provides recommendations for selecting components and connecting interfaces, dealing with special pins, and some layout guidance.

Unused interfaces should be terminated with pull-up or pull-down resistors as indicated in this datasheet or reference schematic. Note that some unused interfaces must be left open. Do not attach pull-up or pull-down resistors to any balls identified as No Connect or Reserved No Connect. There also are reserved pins, identified by RSVD_1P2 and RSVD_VSS that need pull-up or pull-down resistors connected to them. The device can enter special test modes unless these strappings are in place.

8.1 Connecting the PCIe interface

The controller connects to the host system using a PCIe interface which can be configured to operate in several link modes. These are detailed in the functional description. A link between the ports of two devices is a collection of lanes. Each lane has to be AC-coupled between its corresponding transmitter and receiver; with the AC-coupling capacitor located close to the transmitter side (within 1 inch). Each end of the link is terminated on the die into nominal 100Ω differential DC impedance. Board termination is not required.

For information on PCIe, refer to the *PCI Express* Base Specification, Revision 2.0* and *PCI Express* Card Electromechanical Specification, Revision 2.0.*

8.1.1 Link Width Configuration

The device supports a maximum link width of x8, x4, x2, or x1 as determined by the EEPROM LANE_WIDTH field in the PCIe init configuration. This is loaded into the Maximum Link Width field of the PCIe capability Register (LCAP[11:6]; with the silicon default of a x8 link).

During link configuration, the platform and the controller negotiate on a common link width. In order for this to work, the chosen maximum number of PCIe lanes have to be connected to the host system.

8.1.2 Polarity Inversion and Lane Reversal

To ease routing, board designers have flexibility to use the different lane reversal modes supported by the 82598. Polarity inversion can also be used since the polarity of each differential pair is detected during the link training sequence.

When lane reversal is used, some of the down-shift options are not available. For a detailed description of the available combinations, consult the functional description.

8.1.3 PCIe Reference Clock

The device requires a 100 MHz differential reference clock, denoted PE_CLK_P and PE_CLK_N. This signal is typically generated on the system board and routed to the PCIe port. For add-in cards, the clock will be furnished at the PCIe connector.

The frequency tolerance for the PCIe reference clock is +/- 300 ppm.

8.1.4 Bias Resistor

For proper biasing of the PCIe analog interface, a 1.40 KΩ 1% resistor needs to be connected between the PE_RCOMP_P and PE_RCOMP_N pins. To avoid noise coupled onto this reference signal, place the bias resistor close to the controller chip and keep traces as short as possible.

8.1.5 Miscellaneous PCIe Signals

The Ethernet controller signals power management events to the system by pulling low the PE_WAKE# signal. This signal operates like the familiar PCI PME# signal. Somewhere in the system, this signal has to be pulled high to the auxiliary 3.3 V dc supply rail.

The PE_RST# signal, which serves as the familiar reset function for the controller, needs to be connected to the host system's corresponding signal.

8.2 Connecting the MAUI Interfaces

The controller has two High Speed Network Interfaces which can be configured in different 1 and 10 Gb/s operation modes: BX, CX4, KX, KX4, XAUI. Choose the appropriate configuration for your environment.

8.3 MAUI Channels Lane Connections

For BX and KX connections, only the first lane has to be connected (TXx_L0_P, TXx_L0_N; RXx_L0_P, RXx_L0_N). For the rest of the interfaces, all four differential pairs have to be connected per each direction.

These signals are 100 Ω terminated differential signals that are AC coupled near the receiver. Place the AC coupling caps less than 1 inch away from the receiver. For recommended capacitor values, consult the IEEE 802.3 specifications. Capacitor size should be small to reduce parasitic inductance. Use X5R or X7R, ±10% capacitors in a 0402 or 0201 package size.

8.3.1 Bias Resistor

For proper biasing of the MAUI analog interface a 6.49 KΩ 1% resistor needs to be connected between the RBIAS and ground. To avoid noise coupled onto this reference signal, place the bias resistor close to the controller chip and keep traces as short as possible.

8.3.2 XAUI, KX/KX4, CX4 and BX Layout Recommendations

This section provides recommendations for routing high-speed interface. The intent is to route this interface optimally using FR4 technology. Intel has tested and characterized these recommendations.

8.3.2.1 Board Stack Up Example

Printed circuit boards for these designs typically have six, eight, or more layers. Although, the 82598 does not dictate stackup, the following examples are of typical stackups.

Microstrip Example:

- Layer 1 is a signal layer.
- Layer 2 is a ground layer.
- Layer 3 is used for power planes.
- Layer 4 is a signal layer. (Careful routing is necessary to prevent cross talk with layer 5.)
- Layer 5 is a signal layer. (Careful routing is necessary to prevent cross talk with layer 4.)
- Layer 6 is used for power planes.
- Layer 7 is a signal ground layer.
- Layer 8 is a signal layer.

Note: Layer 4 and 5 should be used mostly for low-speed signals because they are referenced to potentially noisy power planes which might also be slotted.

Stripline Example:

- Layer 1 is a signal layer.
- Layer 2 is a ground layer.
- Layer 3 is a signal layer.
- Layer 4 is used for power planes
- Layer 5 is used for power planes
- Layer 6 is a signal layer.
- Layer 7 is a signal ground layer.
- Layer 8 is a signal layer.

Note: To avoid the effect of the potentially noisy power planes on the high-speed signals, use offset stripline topology. The dielectric distance between the power plane and signal layer should be three times the distance between ground and signal layer.

This board stack up configuration can be adjusted to conform to your company's design rules.

8.3.2.2 Trace Geometries

Two types of traces are included: Microstrip or Stripline. Stripline is the preferred solution. Stripline transmission line environments offer advantages that improve performance. Microstrip trace geometries can be used successfully, but it is our recommendation that Stripline geometries be followed.

The following table highlights the height pair-to-pair spacing differences that are recommended between Stripline and Microstrip geometries.

Table 8-1. Pair-to-Pair Spacing

Type	Differential Pair Skew	Differential Pair-to-Pair Spacing	Breakout Length (routes signals from under package)	Lane-to-Lane Skew	XAUI (maximum trace length)†
Microstrip	<5 mils	7 x *h*; where h=dielectric height to closest plane	<200 mils	100 mils	50 cm
Stripline	<5 mils	6 x h; where h=dielectric height to closest plane	<200 mils	100 mils	50 cm

† Typical routing requirement over FR4 material. Recommend 0402 capacitor package size for AC coupling. All other XAUI traces should meet the same spacing requirements as Stripline.

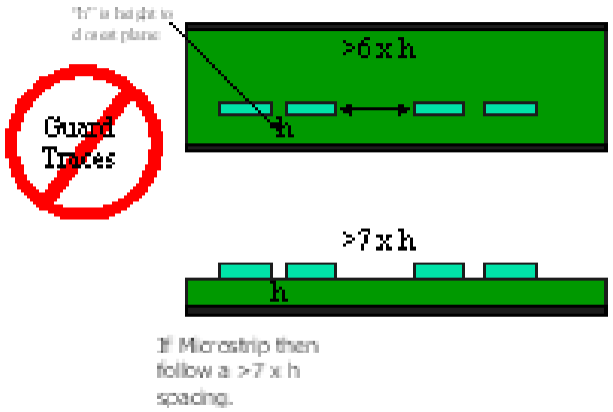


Figure 8-1. Differential Pair Spacing

8.3.2.3 Other High-Speed Signal Routing Practices

These generic layout and routing recommendations are applicable for the MAUI interfaces for the 82598.

In order to keep impedance continuity consistent around via antipad regions, Intel recommends adding the antipad diameter requirement of ≥10 mils clearance to vias to GND and PWR. This ensures that the impedance variance is minimized.

Enforce differential symmetry, even for grounds. Along with ensuring that the MAUI interface is routed symmetrically in terms of signal routing and balance, we also recommended that GND paths are be routed symmetrically. This helps to reduce the imbalance that can occur in the different return current paths.

For the signal trace between the via and AC coupling capacitors on the MAUI interface, there is an intrinsic impedance mismatch because of required capacitors. To minimize the overall effect of having vias and AC coupling capacitors, it is recommended that both the via and capacitor layout pad be placed within 100 mils of each other.

It is best to use a 0402 capacitor or smaller for the AC coupling components on the MAUI interface. The pad geometries for an 0402 or smaller components lend themselves to maintaining a more consistent transmission line environment.

Use smallest possible vias on board to optimize the impedance for the MAUI interface.

8.3.2.4 Via Usage

Use vias to optimize signal integrity. Figure 8-2 shows correct via usage. Figure 8-3 shows the type of topology that should be avoided and can be easily avoided in the board design.

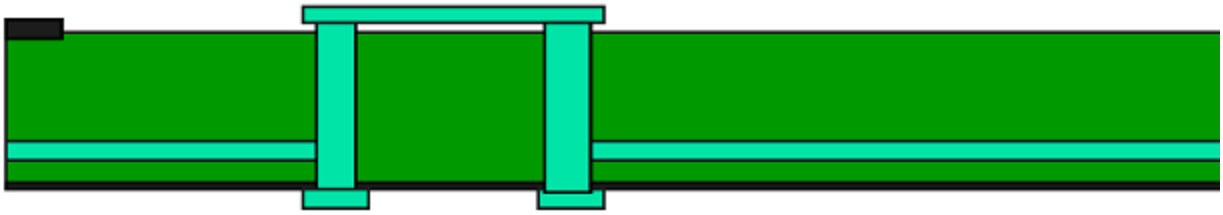


Figure 8-2. Correct Via Usage

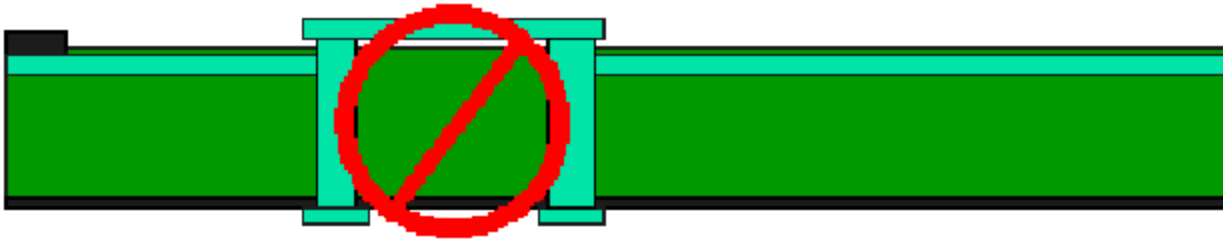


Figure 8-3. Incorrect Via Usage

Place ground vias adjacent to signal vias used for the MAUI interface. Do NOT embed vias between the high-speed signals, but place them adjacent to the signal vias. This helps to create a better GND path for the return current of the AC signals, which in turn helps address impedance mismatches and EMC performance.

We recommend that, in the breakout region between the via and the capacitor pad, you target a Z0 for the via to capacitor trace equal to 50 Ω. This minimizes impedance imbalance.



Figure 8-4. No Vias Between High-Speed Traces

8.3.2.5 Reference Planes

Do not cross plane splits with the MAUI high-speed differential signals. This causes impedance mismatches and negatively affects the return current paths for the board design and layout. Refer to Figure 8-5.

Traces should not cross POWER or GND plane splits if at all possible. Traces should stay 6x the dielectric height away from plane splits or voids. If traces must cross splits, capacitive coupling should be added.

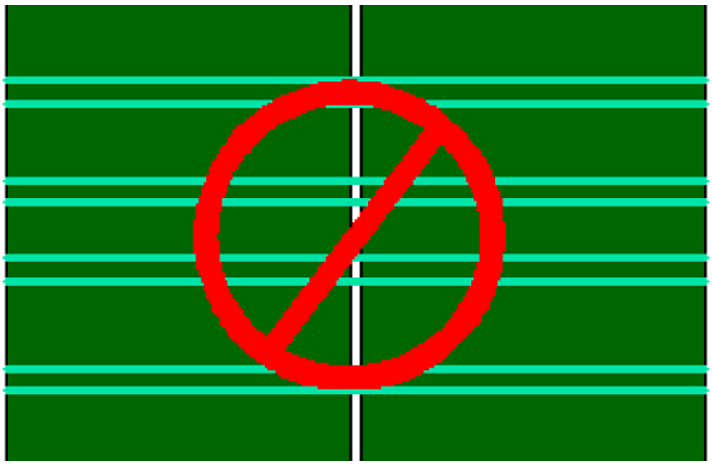


Figure 8-5. Do Not Cross Plane Splits

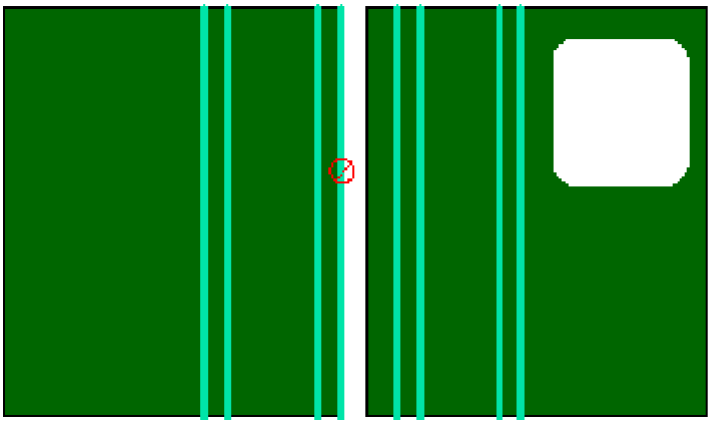
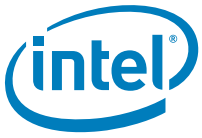


Figure 8-6. Traces 6x Dielectric Splits

Keep Rx and Tx separate. This helps to minimize crosstalk effects since the TX and RX signals are NOT synchronous. This is the more "natural" routing method and will occur without much user interference.

It is also recommended that the MAUI signals stay at least 6x dielectric height away from any POWER or GND plane split. This improves impedance balance and return current paths.

If a high-speed signal needs to reference a power plane, then ensure that the height of the secondary (power) reference plane is at least 3 x h (height) of the primary (ground) reference plane.



8.3.2.6 Dielectric Weave Compensation

Intel recommends slight variations for trace routing to cross the fiberglass weave (or, if traces must be straight for most of their length, rotate the CAD artwork by 15°).

8.3.2.7 Impedance Discontinuities

Impedance discontinuities cause unwanted signal reflections. Minimize vias (signal through holes) and other transmission line irregularities. A total of six through holes (a combination of vias and connector through holes) between the two chips connected by the MAUI interface is a reasonable maximum budget per differential trace. Unused pads and stub traces should also be avoided.

8.3.2.8 Reducing Circuit Inductance

Traces should be routed over a continuous reference plane with no interruptions. If there are vacant areas on a reference or power plane, the signal conductors should not cross the vacant area. This causes impedance mismatches and associated radiated noise levels. Noisy logic grounds should NOT be located near or under high-speed signals or near sensitive analog pin regions of the LAN silicon. If a noisy ground area must be near these sensitive signals or IC pins, ensure sufficient decoupling and bulk capacitance in these areas. Noisy logic and switching power supply grounds can sometimes affect sensitive DC subsystems such as analog to digital conversion, operational amplifiers, etc. All ground vias should be connected to every ground plane; and similarly, every power via, to all power planes at equal potential. This helps reduce circuit inductance. Another recommendation is to physically locate grounds to minimize the loop area between a signal path and its return path. Rise and fall times should be as slow as possible. Because signals with fast rise and fall times contain many high frequency harmonics, which can radiate significantly. The most sensitive signal returns closest to the chassis ground should be connected together. This will result in a smaller loop area and reduce the likelihood of crosstalk. The effect of different configurations on the amount of crosstalk can be studied using electronics modeling software.

8.3.2.9 Signal Isolation

To maintain best signal integrity, keep digital signals far away from the analog traces. A good rule of thumb is no digital signal should be within 7x to 10x dielectric height of the differential pairs. If digital signals on other board layers cannot be separated by a ground plane, they should be routed at a right angle (90 degrees) to the differential pairs. If there is another LAN controller on the board, take care to keep the differential pairs from that circuit away. Same thing applies to switching regulator traces.

Some rules to follow for signal isolation:

- Separate and group signals by function on separate layers if possible. If possible, maintain a gap of 7x to 10x dielectric height between all differential pairs (Ethernet) and other nets, but group associated differential pairs together (Example: Tx with Tx and Rx with Rx).
- Over the length of the trace run, each differential pair should be at least 7x to 10x dielectric height away from any parallel signal traces.
- Physically group together all components associated with one clock trace to reduce trace length and radiation.
- Isolate other I/O signals from high-speed signals to minimize crosstalk, which can increase EMI emission and susceptibility to EMI from other signals.
- Avoid routing high-speed LAN traces near other high-frequency signals associated with a video controller, cache controller, processor, or other similar devices.

8.3.2.10 Power and Ground Planes

Good grounding requires minimizing inductance levels in the interconnections and keeping ground returns short, signal loop areas small, and locating decoupling capacitors at or near power inputs to bypass to the signal return. This 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 will increase inductance and increase EMI radiation levels.
- Use distance and/or extra decoupling capacitors to 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 will minimize 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.
- Do not route high-speed signals near switching regulator circuits.

8.4 Connecting the Serial EEPROM

The controller uses an Serial Peripheral Interface (SPI)* EEPROM. Several words of the EEPROM are accessed automatically by the device after reset to provide pre-boot configuration data before it is accessed by host software. The remainder of the EEPROM space is available to software for storing the MAC address, serial numbers, and additional information. For a complete description of the content stored in the EEPROM please consult the NVM Map section of this document.

8.4.1 Supported EEPROM devices

Table 8-2 lists the SPI EEPROMs that have been found to work satisfactorily with the 82598 device. The SPI EEPROMs used must be rated for a clock rate of at least 2 MHz.

Table 8-2. Supported SPI EEPROM Devices

Density [Kb]	Atmel PN	STM PN	Catalyst PN
4	AT25040AN-10SI-2.7	M95040WMN6T	CAT25C040S-TE13
8	AT25080AN-10SI-2.7	M95080WMN6T	CAT25C080S-TE13
16	AT25160AN-10SI-2.7	M95160WMN6T	CAT25C16S-TE13
32	AT25320AN-10SI-2.7	M95320WMN6T	CAT25C32S-TE13
64	AT25640AN-10SI-2.7	M95640WMN6T	CAT25C64S-TE13
128	AT25128AN-10SI-2.7	M95128WMN6T	CAT25CS128-TE13
256	AT25256AN-10SI-2.7	M95256WMN6T	

Use a 128 Kb EEPROM for all applications until an appropriate size for each application is determined. Recommended manufacturer and part numbers are Atmel's AT25128N or Microchip's 25LC128.

For more information on how to properly attach the EEPROM device to the 82598 controller please follow the example provided in the Reference Schematic.

If no EEPROM device is used leave EE_CE_N, EE_SCK, EE_SI, EE_SO unconnected.

8.4.2 EEUPDATE

Intel has an MS-DOS* software utility called EEUPDATE, which can be used to program EEPROM images in development or production environments. To obtain a copy of this program, contact your Intel representative.

8.5 Connecting the Flash

The controller provides support for an SPI Flash device which will be made accessible to the system through the following options:

- The Flash Base Address register (PCIe Control register at offset 0x14 or 0x18).
- An address range of the IOADDR register, defined by the IO Base Address register (PCIe) Control register at offset 0x18 or 0x20).
- The Expansion ROM Base Address register (PCIe Control register at offset 0x30).

8.5.1 Supported EEPROM Devices

The controller supports SPI flash type. All supported Flashes have address size of 24 bits. TTable 8-3 lists the specific Flash types that are supported.

Table 8-3. Supported Flash Types

Density	Atmel PN	STM PN
512KBit	AT25F512N-10SI-2.7	M25P05-AVMN6T
1MBit	AT25F1024N-10SI-2.7	M25P10-AVMN6T
2MBit	AT25F2048N-10SI-2.7	M25P20-AVMN6T
4MBit	AT25F4096N-10SI-2.7	M25P40-AVMN6T
8MBit		M25P80-AVMN6T
16MBit		M25P16-AVMN6T
32Mbit		M25P32-AVMN6T

For more information on how to properly attach the Flash device to the controller, follow the example provided in the Reference Schematic.

If no Flash device is used leave FLSH_CE_N, FLSH_SCK, FLSH_SI, FLSH_SO unconnected.

8.6 Connecting the Manageability Interfaces

SMBus and NC-SI are optional interfaces for pass-through and/or configuration traffic between the BMC and the 82598.

Note: Intel recommends that the SMBus be connected to the MCH or BMC for the EEPROM recovery solution. If the connection is to a BMC, it is able to send the EEPROM release command.

The 82598 can be connected to an external BMC and can operate in one of two modes:

- SMBus mode
- NC-SI mode

Note: Clock out (if enabled) is provided in all power states (unless the 82598 is disabled).

For more information on system management solutions, refer to Section 5..

8.6.1 Connecting the SMBus Interface

To connect the SMBus interface to the chipset or the BMC, connect the SMBD, SMBCLK and SMBALRT_N signals to the corresponding pins of the chipset/BMC (see Figure 8-1). Note that these pins require pull-up resistors to the 3.3 V dc supply rail.

If the SMBus interface is not used, the previously mentioned pull-up resistors on the SMBD, SMBCLK and SMBALRT_N signals still have to be in place.

8.6.2 Connecting the NC-SI Interface

The NC-SI interface is a connection to an external BMC. It operates as a single interface with an external BMC, where all traffic between the 82598 and the BMC flows through the interface (see Figure 8-1).

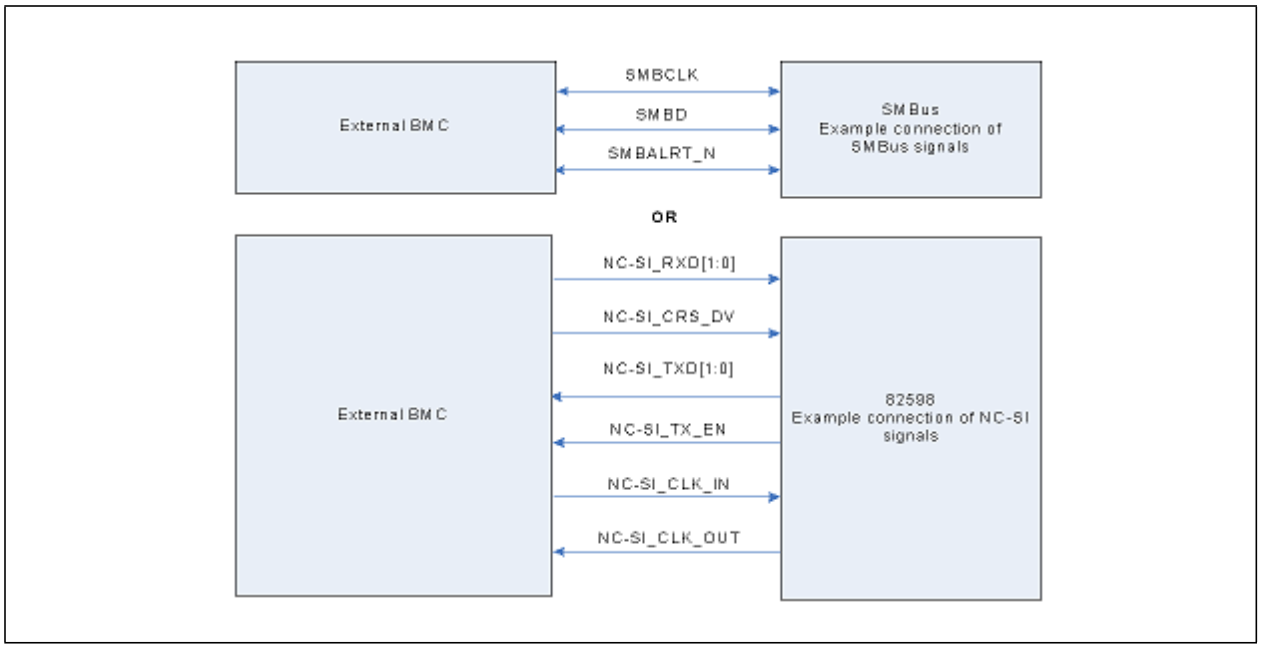


Figure 8-1. External BMC Connections with NC-SI and SMBus

8.6.3 NC-SI Electrical Interface Requirements

This section describes the hardware implementation requirements necessary to meet the NC-SI physical layer standard. Board-level design requirements are included for connecting the 82598 Ethernet solution to an external BMC. The layout and connectivity requirements are addressed in low-level detail. This section, in conjunction with the *Network Controller Sideband Interface Specification Version 0.8 NC-SI Specification*, also provides the complete board-level requirements for the NC-SI solution.

The 82598's on-board SMBus port enables network manageability implementations required for remote control and alerting via the LAN. With SMBus, management packets can be routed to or from a BMC. Enhanced pass-through capabilities also enable system remote control over standardized interfaces. Also included is a new manageability interface (NC-SI) that supports the DMTF preOS sideband protocol. An internal management interface called MDIO enables the MAC (and software) to monitor and control the PHY.

8.6.3.1 External Baseboard Management Controller (BMC)

The external BMC is required to meet the latest NC-SI specification as it relates to the RMII electrical interface.

8.6.3.2 NC-SI Reference Schematic

Figure 8-2 shows the recommended pull-up and pull-downs that should be used on the NC-SI interface regardless of the application, even when not using the interface.

- NCSI_CLK_IN: A pull down should always be placed on this input to the 82598.
- NCSI_CRS_DV: A pull-down should always be used to ensure the data valid is never indicated during start-up when the signal is not driven.

- NCSI_RXD_0/1: Pull-up resistors should always be used to ensure that these lines stay at a logic high when nothing is driven, the interface is not used, and for multiple drop configurations.
- NCSI_TX_EN: A pull-down should always be used to ensure the Tx is never enabled during start-up or when the signal is not driven.
- NCSI_TXD_0/1: Pull-up resistors should always be used to ensure that these lines stay at a logic high when nothing is driven, the interface is not used, and for multiple drop configurations.

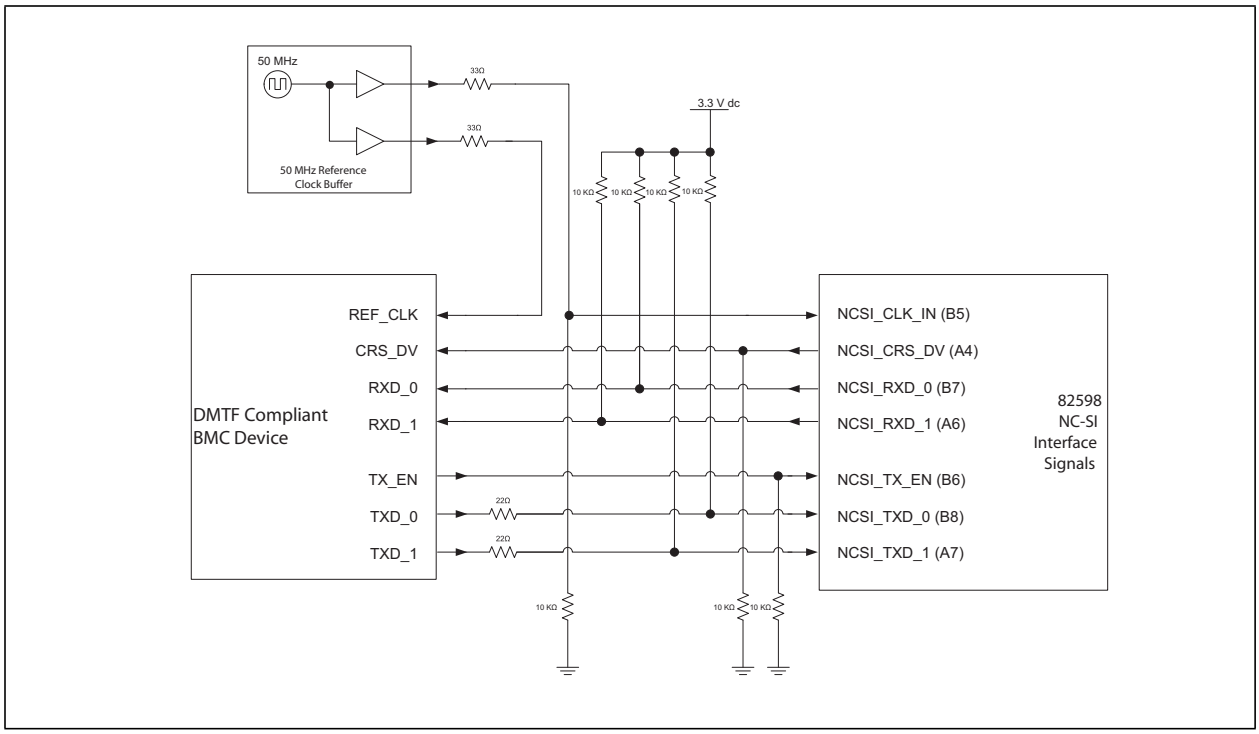
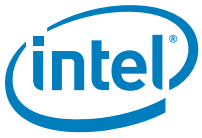


Figure 8-2. NC-SI Connection Requirement - Single Drop Configuration



8.7 Resets

After power is applied to the 82598 controller, it must be reset. There are two ways to do this: (1) using the Internal Power On Reset circuit and (2) using the external LAN_PWR_GOOD signal. By default, Internal Power On Reset will reset the chip. If the design relies on Internal Power On Reset, then the power supply sequencing timing requirement (see Section 7.4.2) between the 1.8 V dc and 1.2 V dc power rails has to be met. If this requirement is impossible to meet, the alternative is to bypass the Internal Power On Reset circuit by pulling POR_BYPASS high and using an external power monitoring solution to provide a LAN_PWR_GOOD signal. For LAN_PWR_GOOD timing requirements, see Section 7.6.6.

Table 8-4. Reset Context for POR_BYPASS and LAN_PWR_GOOD

POR_BYPASS	Active Reset Circuit	
If = 0b	LAN_PWR_GOOD	
If = 1b	External Reset LAN_PWR_GOOD	
	If = 0b	Held in reset
	If = 1b	Initialized, ready for normal operation

It is important to ensure that resets for the BMC and the 82598 are generated within a specific time interval. The NC-SI specification calls out a requirement of link establishment in two seconds of the BMC receiving the power good signal from the platform.

To achieve link within the time specified, the 82598 and external BMC need to receive power good signals from the platform within one second of each other. This causes an Internal Power On Reset; the NC-SI interface is also reset, ready to link. The BMC should poll this interface and to establish link for two seconds to ensure compliance.

8.8 NC-SI Layout Requirements

8.8.1 Board Impedance

The NC-SI manageability interface is a single ended signaling environment and as such we recommend a target board and trace impedance of 50 Ω plus 20% and minus 10%. This ensures optimal signal integrity.

8.8.2 Trace Length Restrictions

We recommend a trace length maximum value from a board placement and routing topology perspective of eight inches for direct connect applications. This ensures signal integrity and quality is preserved from a design perspective and that compliance is met for NC-SI electrical requirements.

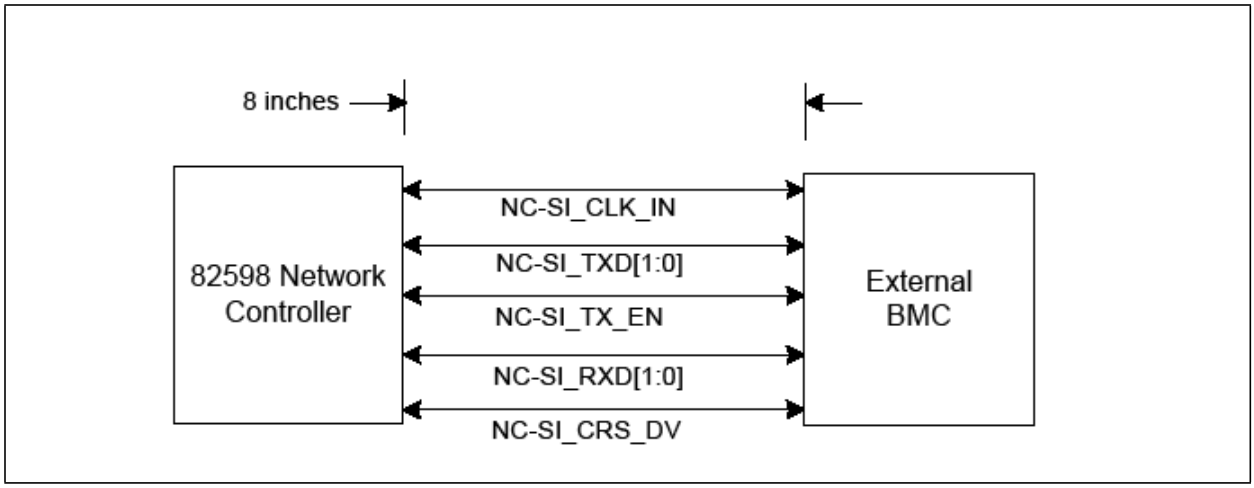


Figure 8-7. NC-SI Maximum Trace Length Requirement Between the 82598 and External BMC for Direct Connect Applications.

For multi-drop applications, we have a spacing recommendation of maximum four inches between the two packages connected to the same BMC. This is done to keep the overall length between the BMC and the network controller within specification.

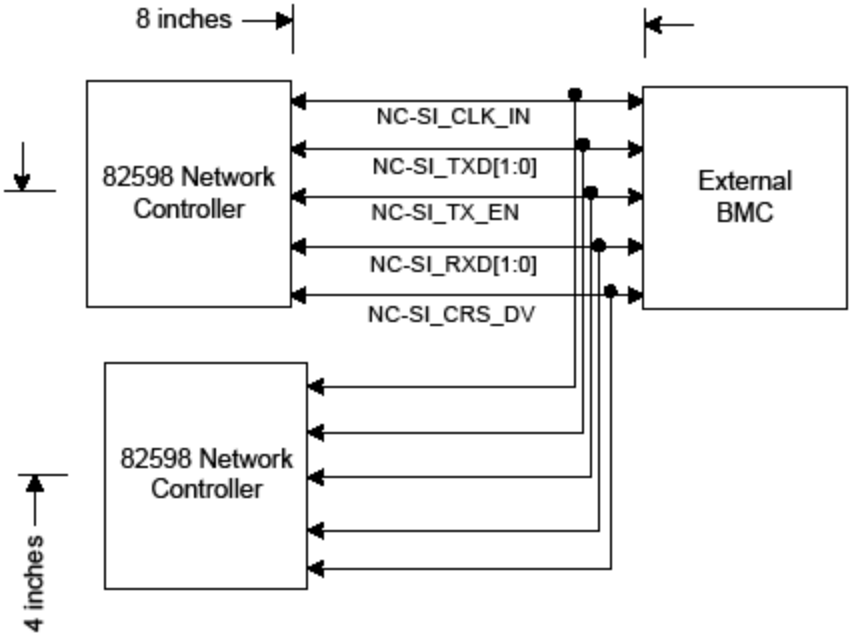


Figure 8-8. NC-SI Maximum Trace Length Requirement Between the 82598 and External BMC for Multi-Drop Applications

8.8.3 Special Delay Requirements

The 82598 controller violates the DMTF NC-SI AC timing specification in terms of the hold (Thd) time and minimum clock to output timing (Tco min), and needs special attention during the layout design.

To ensure the controller will be able to communicate with a specification-compliant BMC the following guidelines must be followed:

- The clock traces from the clock source to the BMC and the one from the source to the 82598 have to be length matched within 5 mils.
- Make sure there the total skew between the clocks at the input of the BMC and the 82598 is less than 1 ns. This 1 ns should include the skew introduced by the clock buffer and by the difference of input capacitance of the clock input buffers on the BMC and the 82598 respectively, as well as any skew introduced by clock trace length differences.
- The delay introduced by the transmit and receive data lines should be equal, and not less than 0.3 ns. This translates to roughly two inches assuming 150 ps/inch propagation delay.

8.9 Connecting the MDIO Interfaces

The 82598 provides one MDIO interface for each LAN port to be used as configuration interface for an external PHY attached to the controller.

Connect the MDIO and MDC signals to the corresponding pins on the PHY chip. Please make sure to provide a pull up to 3.3 V dc on the MDIO signal.

8.10 Connecting the Software-Definable Pins (SDPs)

The controller has eight software-defined pins (SDP) per port that can be used for miscellaneous hardware or software-controllable purposes. These pins and their function are bound to a specific LAN device. These pins can each be individually configured to act as either input or output pins via EEPROM. The initial value in case of an output can also be configured in the same way, however the silicon default for any of these pins is to be configured as outputs.

To avoid signal contention, all eight pins are set as input pins until after EEPROM configuration has been loaded.

Choose the right software definable pins for your applications keeping in mind that two of the eight pins: SDPx_6 and SDPx_7 are open drain, the rest are tri-state buffers. Also take in consideration that four of these pins (SDPx_0 – SDPx_3) can be used 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 62.5 MHz, as opposed to an edge-detection circuit). When detected, a corresponding GPI interrupt is indicated in the Interrupt Cause register.

When connecting the software definable pins to different digital signals please keep in mind that these are 3.3 V signals and use level shifting if necessary.

The use, direction, and values of SDPs are controlled and accessed using fields in the Extended SDP Control (ESDP) and Extended OD SDP Control (EODSDP).

8.11 Connecting the Light Emitting Diodes for Designs Based on the 82598 Controller

The 82598 controller provides four programmable high-current push-pull (active high) outputs per port to directly drive LEDs for link activity and speed indication. Each LAN device provides an independent set of LED outputs; these pins and their function are bound to a specific LAN device. Each of the four LED outputs can be individually configured to select the particular event, state, or activity, which will be 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 LED ports are fully programmable through the EEPROM interface, LEDCTL register. In addition, the hardware-default configuration for all LED outputs can be specified via an EEPROM field, thus supporting LED displays configurable to a particular OEM preference.

Please provide a separate current limiting resistors for each LED connected. Since the LEDs are likely to be placed close to the board edge and to external interconnects, take care to route the LED traces away from potential sources of EMI noise. In some cases, it may be desirable to attach filter capacitors.

8.12 Connecting the Miscellaneous Signals

8.12.1 LAN Disable

The 82598 has two signals that can be used for disabling Ethernet functions from system BIOS. LAN0_DIS_N and LAN1_DIS_N are the separated port disable signals. Each signal can be driven from a system output port. Choose outputs from devices that retain their values during reset. For example, some ICH GPIO outputs transition high during reset. It is important not to use these signals to drive LAN0_DIS_N or LAN1_DIS_N because these inputs are latched upon the rising edge of PE_RST_N or an in-band reset end.

Each PHY may be disabled if its LAN function's LAN Disable input indicates that the relevant function should be disabled. Since the PHY is shared between the LAN function and manageability, it may not be desirable to power down the PHY in LAN Disable. The PHY_in_LAN_Disable EEPROM bit determines whether the PHY (and MAC) are powered down when the LAN Disable pin is asserted. Default is not to power down.

A LAN port can also be disabled through EEPROM settings. If the LAN_DIS EEPROM bit is set, the PHY enters power down. Note, however, that setting the EEPROM LAN_PCI_DIS bit does not bring the PHY into power down.

Table 8-5. PCI/LAN Function Index

PCI Function #	LAN Function Select	Function 0	Function 1
Both LAN functions are enabled	0	LAN 0	LAN 1
LAN 0 is disabled	0	Dummy	LAN1
LAN 1 is disabled	0	LAN 0	-
LAN 0 is disabled	1	LAN 1	-
Both LAN functions are enabled	1	LAN 1	LAN 0
LAN 1 is disabled	1	Dummy	LAN 0
Both LAN functions are disabled	Don't Care	All PCI functions are disabled Whole Device is at deep PD	

When both LAN ports are disabled following a Power on Reset / LAN_PWR_GOOD/ PE_RST_N/ In-Band reset the LANx_DIS_N signals should be tied statically to low. At this state the device is disabled, LAN ports are powered down, all internal clocks are shut and the PCIe connection is powered down (similar to L2 state).

8.12.2 BIOS Handling of Device Disable

Assume that in the following power up sequence the LANx_DIS_N signals are driven high (or it is already disabled):

1. PCIe link is established following the PE_RST_N.
2. BIOS recognizes that the device should be disabled.
3. BIOS drives the LANx_DIS_N signals to the low level.
4. BIOS issues PE_RST_N or an In-Band PCIe reset.
5. As a result, the device samples the LANx_DIS_N signals and enters the desired device-disable mode.
6. Re-enable could be done by driving high one of the LANx_DIS_N signals and then issuing a PE_RST_N to restart the device.

8.12.3 PHY Disable and Device Power Down Signals

The controller has two signals dedicated for powering down a connected external PHY device. These signals (PHYx_PWRDN_N) can be connected to the power down/disable input of the external PHY or can be left unconnected.

The controller also provides a DEV_PWRDN_N output signal that can be used to control the power delivery circuit for the chip based on its internal power state.

For detailed operation of these three signals please consult the Functional Description chapter.

8.13 Oscillator Design Considerations

This section provides information regarding oscillators for use with the 82598 controller.

All designs require a 156.25 MHz external clock source. The 82598 uses this 156.25 MHz source to generate clocks with frequency up to 3.125 GHz for the high speed interfaces.

The chosen oscillator vendor should be consulted early in the design cycle. Oscillator manufacturers familiar with networking equipment clock requirements may provide assistance in selecting an optimum, low-cost solution.

8.13.1 Oscillator Types

8.13.1.1 Fixed Crystal Oscillator

A packaged fixed crystal oscillator comprises an inverter, a quartz crystal, and passive components. The device renders a consistent square wave output. Oscillators used with microprocessors are supplied in many configurations and tolerances.

Crystal oscillators can be used in special situations, such as shared clocking among devices. As clock routing can be difficult to accomplish, it is preferable to provide a separate crystal for each device.

For Intel controllers, it is acceptable to overdrive the internal inverter by connecting a 156.25 MHz external oscillator to REFCLKIN_P and REFCLKIN_N leads. The oscillator should be specified to drive CMOS logic levels, and the clock trace to the device should be as short as possible. The chosen device specifications should call for a 45% (minimum) to 55% (maximum) duty cycle and a ±50 ppm frequency tolerance.

8.13.1.2 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.

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, use of programmable oscillators is generally not recommended.

8.13.2 Oscillator Solution

Choose a clock oscillator with LVPECL output. When connecting the output of the oscillator to an 82598 controller, use AC coupling. 100nF is reasonable value to use. To avoid unwanted reflections on the clock signal which can lead to non monotonic clock edges, make sure that the transmission line is correctly terminated. A termination example is shown in Figure 8-5.

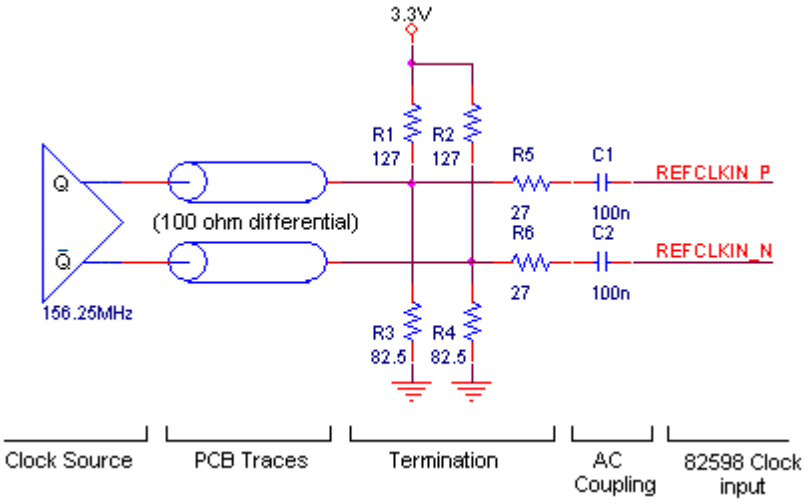


Figure 8-9. Reference Oscillator Circuit

Note: The input clock jitter from the oscillator can impact the 82598 clock and its performance. If output jitter performance is poor, a lower jitter clock oscillator should be chosen.

Recommended oscillators are:

Table 8-6. Part Numbers for Recommended Oscillators

Valpey Fisher	VF266B-156.250MHZ
Pletronics Inc.	PE7744DV-156.25M
Pericom Semiconductor Corporation	SEL3833B-156.2500T
MMD Components	MI3050LAT3-156.250MHZ-T

8.13.3 Oscillator Layout Recommendations

Oscillators should not be placed near I/O ports or board edges. Noise from these devices may be coupled onto the I/O ports or out of the system chassis. Oscillators should also be kept away from XAUI differential pairs to prevent interference.

The reference clock should be routed differentially; use the shortest, most direct traces possible. Keep potentially noisy traces away from the clock trace. It is critical to place the termination resistors and AC coupling capacitors as close to the 82598 as possible (less than 250 mils).

8.13.4 Reference Clock Measurement Recommendations

A low capacitance, high impedance probe (C < 1 pF, R > 500 KΩ) should be used for testing. Probing parameters can affect the measurement of the clock amplitude and cause errors in the adjustment. A test should be done after the probe has been removed to ensure circuit operation.

8.14 Power Supplies

The controller requires three power rails: 3.3 V dc, 1.8 V dc and 1.2 V dc. A central power supply can provide all the required voltage sources; or power can be derived from the 3.3 V dc supply and regulated locally using external regulators. If the LAN wake capability is used, voltages must remain present during system power down. Local regulation of the LAN voltages from system 3.3 Vmain and 3.3 Vaux voltages is recommended.

Make sure that all the external voltage regulators generate the proper voltage, meet the output current requirements (with adequate margin), and provide the proper power sequencing. For details, see the electrical specification section of this document.

8.14.1 Power Supply Sequencing

Due to the current demand, a Switching Voltage Regulator (SVR) is highly recommended for the 1.2 V dc power rail. Regardless of the type of regulator used, all regulators need to adhere to the sequencing shown in the following figure to avoid latch-up and forward-biased internal diodes (1.8 V dc must not exceed 3.3 V dc; 1.2 V dc must not exceed 3.3 V dc; 1.2 V dc must not exceed 1.8 V dc).

The power supplies are all expected to ramp during a short power-up internal (recommended interval 20 ms or quicker). The delay between the 1.8 V dc and 1.2 V dc rail, measured at a level of 400 mV, has to be quicker than 500 µs. Do not leave the device in a prolonged state where some, but not all, voltages are applied.

8.14.1.1 Using Regulators With Enable Pins

The use of regulators with enable pins is very helpful in controlling sequencing. Connecting the enable of the 1.8 V dc regulator to 3.3 V dc will allow the 1.8 V dc to ramp. Connecting the enable of the 1.2 V dc regulator to the 1.8 V dc output assures that the 1.2 V dc rail will ramp after the 1.8 V dc rail. This provides a quick solution to power sequencing. Make sure to check design parameters for inputs with this configuration. Alternatively power monitoring chips can be used to provide the proper sequencing by keeping the voltage regulators with lower output in shutdown until the one immediately above doesn't reach a certain output voltage level.

8.14.2 Power Supply Filtering

These filters provide several high-frequency bypass capacitors for each power rail. Select values in the range of 0.001 µF to 0.1 µF and, if possible, orient the capacitors close to the device and adjacent to power pads.

Traces between decoupling and I/O filter capacitors should be as short and wide as practical. Long and thin traces are more inductive and would reduce the intended effect of decoupling capacitors. Also for similar reasons, traces to I/O signals and signal terminations should be as short as possible. Vias to the decoupling capacitors should be sufficiently large in diameter to decrease series inductance.

Table 8-7. Minimum Number of Bypass Capacitors per Power Rail.

Power Rail	Total Bulk Capacitance	0.1µF	0.001µF
3.3 V dc	22 µF	5	0
1.8 V dc	66 µF	8	0
1.2 V dc	160 µF	40	6

8.14.3 Support for Power Management and Wake Up

The designer must connect the MAIN_PWR_OK and the AUX_PWR signals on the board. These are digital inputs to the 82598 controller and serve the following purpose:

The MAIN_PWR_OK will signal the 82598 controller that the main power from the system is up and stable. For example it could be pulled up to the 3.3 V dc main rail, or connected to a power well signal available in the system.

When sampled high AUX_PWR will indicate that auxiliary power is available to the controller, and therefore the controller advertises D3cold Wake Up support. The amount of power required for the function (which includes the entire network interface card) is advertised in the Power Management Data Register, which is loaded from the EEPROM.

If wakeup support is desired, AUX_PWR needs to be pulled high and the appropriate wakeup LAN address filters must also be set. The initial power management settings are specified by EEPROM bits. When a wakeup event occurs the controller asserts the PE_WAKEn signal to wake the system up. PE_WAKEn remains asserted until PME status is cleared in the 82598 Power Management Control/ Status Register.

8.15 Connecting the JTAG Port

The 82598 10 Gigabit Ethernet Controller contains a test access port (3.3 V dc only) conforming to the IEEE 1149.1a-1994 (JTAG) Boundary Scan specification. To use the test access port, connect these balls to pads accessible by your test equipment.

For proper operation a pull-down resistor should be connected to the JTCK and JRST_N signals and pull up resistors to the JTMS and JTDI signals.

A BSDL (Boundary Scan Definition Language) file describing the 82598 10 Gigabit Ethernet Controller device is available for use in your test environment. The controller also contains an XOR test tree mechanism for simple board tests. Details of XOR tree operation are available from your Intel representative.

8.16 Thermal Design Considerations

In a system environment, the temperature of a component is a function of both the system and component thermal characteristics. System-level thermal constraints consist of the local ambient temperature at the component, the airflow over the component and surrounding board, and the physical constraints at, above, and surrounding the component that may limit the size of a thermal enhancement (heat sink).

The component's case/die temperature depends on:

- Component power dissipation
- Size
- Packaging materials (effective thermal conductivity)
- Type of interconnection to the substrate and motherboard
- Presence of a thermal cooling solution
- Power density of the substrate, nearby components, and motherboard

All of these parameters are pushed by the continued trend of technology to increase performance levels (higher operating speeds, MHz) and power density (more transistors). As operating frequencies increase and packaging size decreases, the power density increases and the thermal cooling solution space and airflow become more constrained. The result is an increased emphasis on system design to ensure that thermal design requirements are met for each component in the system.

8.16.1 Importance of Thermal Management

The thermal management objective is to ensure system component temperatures are maintained in functional limits. The functional temperature limit is the range in which electrical circuits meet specified performance requirements. Operation outside the functional limit can degrade performance, cause logic errors, or cause system damage.

Temperatures exceeding the maximum operating limits may result in irreversible changes in the device operating characteristics. Note that sustained operation at component maximum temperature limit may affect long-term device reliability.

8.16.2 Packaging Terminology

The following is a list of packaging terminology used in this document.

FCBGA Flip Chip Ball Grid Array: A surface-mount package using a combination of flip chip and BGA structure whose PCB-interconnect method consists of Pb-free solder ball array on the interconnect side of the package. The die is flipped and connected to an organic build-up substrate with C4 bumps.

Junction: Refers to a P-N junction on the silicon. In this document, it is used as a temperature reference point (for example, $_{JA}$ refers to the junction to ambient thermal resistance).

Ambient: Refers to local ambient temperature of the bulk air approaching the component. It can be measured by placing a thermocouple approximately one inch upstream from the component edge.

Lands: Pads on the PCB to which BGA balls are soldered.

PCB: Printed circuit board.

Printed Circuit Assembly (PCA): An assembled PCB.

Thermal Design Power (TDP): Estimated maximum possible/expected power generated in a component by a realistic application.

LFM: Linear feet per minute (airflow).

$_{JA}$ (Theta JA): Thermal resistance junction-to-ambient, °C/W.

$_{JT}$ (Psi JT): Junction-to-top (of package) thermal characterization parameter, °C/W.

8.16.3 Thermal Specifications

To ensure proper operation, the thermal solution must maintain a case temperature at or below the values specified in Table 8-8. System-level or component-level thermal enhancements are required to dissipate the generated heat to ensure the case temperature never exceeds the maximum temperatures. Table 8-9 lists the thermal performance parameters for the JEDEC JESD51-2 standard.

Analysis indicates that real applications are unlikely to cause the 82598 to be at Tcase-max for sustained periods of time. Given that Tcase should reasonably be expected to be a distribution of temperatures, sustained operation at Tcase-max may affect long-term reliability of the system, and sustained performance at Tcase-max should be evaluated during the thermal design process and steps taken to reduce the Tcase temperature.

Good system airflow is critical to dissipate the highest possible thermal power. The size and number of fans, etc. and their placement in relation to components and airflow channels within the system determine airflow.

Table 8-8. Package Thermal Characteristics in Standard JEDEC Environment

Package	(°C/W)	(°C/W)
31 mm FCBGA	14.6[1]	1.6
31 mm FCBGA -HS	11.9[2]	0.6

[1] Integrated Circuit Thermal Measurement Method-Electrical Test Method EIA/JESD51-1, Integrated Circuits Thermal Test Method Environmental Conditions – Natural Convection (Still Air), No Heat sink attached EIAJESD51-2.

[2] Natural Convection (Still Air), Heat sink attached.

Table 8-9. T Estimated Thermal Design Power

82598 Estimated TDP[1]	6.5 W
Tcase Max-hs[2]	106 °C[3]

[1] Power values shown are preliminary and reflect pre-silicon simulation estimates; once silicon becomes available, Maximum power, also known as Thermal Design Power (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.

[2] Tcase Max-hs is defined as the maximum case temperature with the Default Enhanced Thermal Solution attached.

[3] This is a not to exceed maximum allowable case temperature.

The thermal parameters defined above are based on simulated results of packages assembled on standard multi layer 2s2p 1.0-oz Cu layer boards in a natural convection environment. The maximum case temperature is based on the maximum junction temperature and defined by the relationship, maximum Tcase = Tjmax – ($_{JT}$ x Power) where $_{JT}$ is the junction-to-top (of package) thermal characterization parameter. If the case temperature exceeds the specified Tcase max, thermal enhancements such as heat sinks or forced air will be required. $_{JA}$ is the thermal resistance junction-to-ambient of the package.

8.16.3.1 Case Temperature

The 82598 is designed to operate properly as long as the Tcase rating is not exceeded.

8.16.4 Thermal Attributes

8.16.4.1 Typical System Definition

A system with the following attributes was used to generate thermal characteristics data:

- A heatsink case with the Default Enhanced Thermal Solution.
- Four-layer, 4.5 x 4 inch PCB.

8.16.4.2 Package Mechanical Attributes

The 82598 product line is packaged in a 31 mm x 31 mm FCBGA.

8.16.4.3 Package Thermal Characteristics

See Figure 8-10 and Table 8-10 to determine an optimum airflow and heatsink combination. Your system design may vary from the system board environment used to generate the values shown.

Contact your Intel sales representative for product line thermal models.

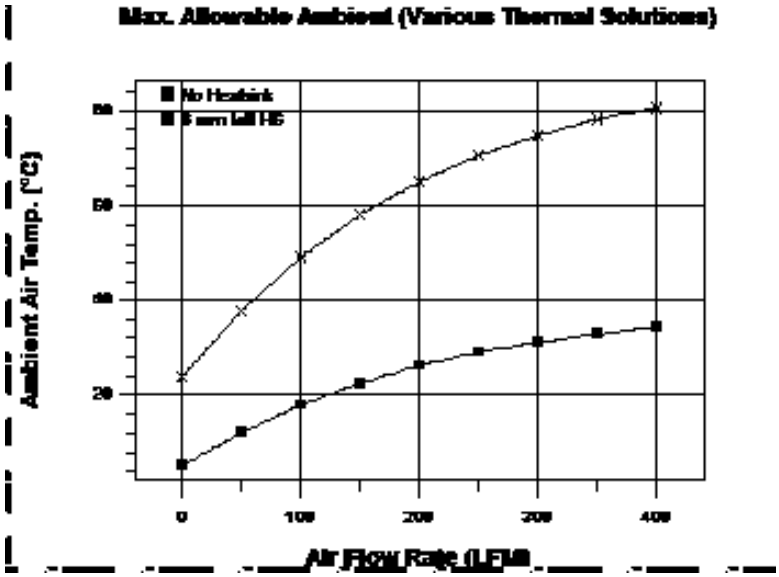


Figure 8-10. 82598 Max Allowable Ambient @ 6.5 W

Table 8-10. 82598 Expected Tcase (°C) for 8.0 mm Heat Sink at 6.5 W

Heatsink Attached									Tcase Max=105°C
85 °C amb.	167	153	142	133	126	120	116	113	110
80 °C amb.	162	148	137	128	121	115	111	108	105
75 °C amb.	157	143	132	123	116	110	106	103	100
70 °C amb.	152	138	127	118	111	105	101	98	95
65 °C amb.	147	133	122	113	106	100	96	93	90
60 °C amb.	142	128	117	108	101	95	91	88	85
55 °C amb.	137	123	112	103	96	90	86	83	80
50 °C amb.	132	118	107	98	91	85	81	78	75
45 °C amb.	127	113	102	93	86	80	76	73	70
AirFlow (LFM)	0	50	100	150	200	250	300	350	400

NOTE: The underlined value(s) indicate airflow/ambient combinations that exceed the allowable case temperature for the the 82598 at 7.2 W.

8.16.5 Thermal Enhancements

One method frequently used to improve thermal performance is to increase the device surface area by attaching a metallic heatsink to the component top. Increasing the surface area of the heatsink reduces the thermal resistance from the heatsink to the air, increasing heat transfer.

8.16.5.1 Clearances

To be effective, a heatsink should have a pocket of air around it that is free of obstructions. Though each design may have unique mechanical restrictions, the recommended clearances for a heatsink used with are shown in Figure 8-11 and Figure 8-12.

The 8.0 mm heatsink is the recommended thermal solution for the 82598 for 6.5 W and lower. However, if sufficient airflow is not available in your system, an active heat sink can be used with the recommended clearances.

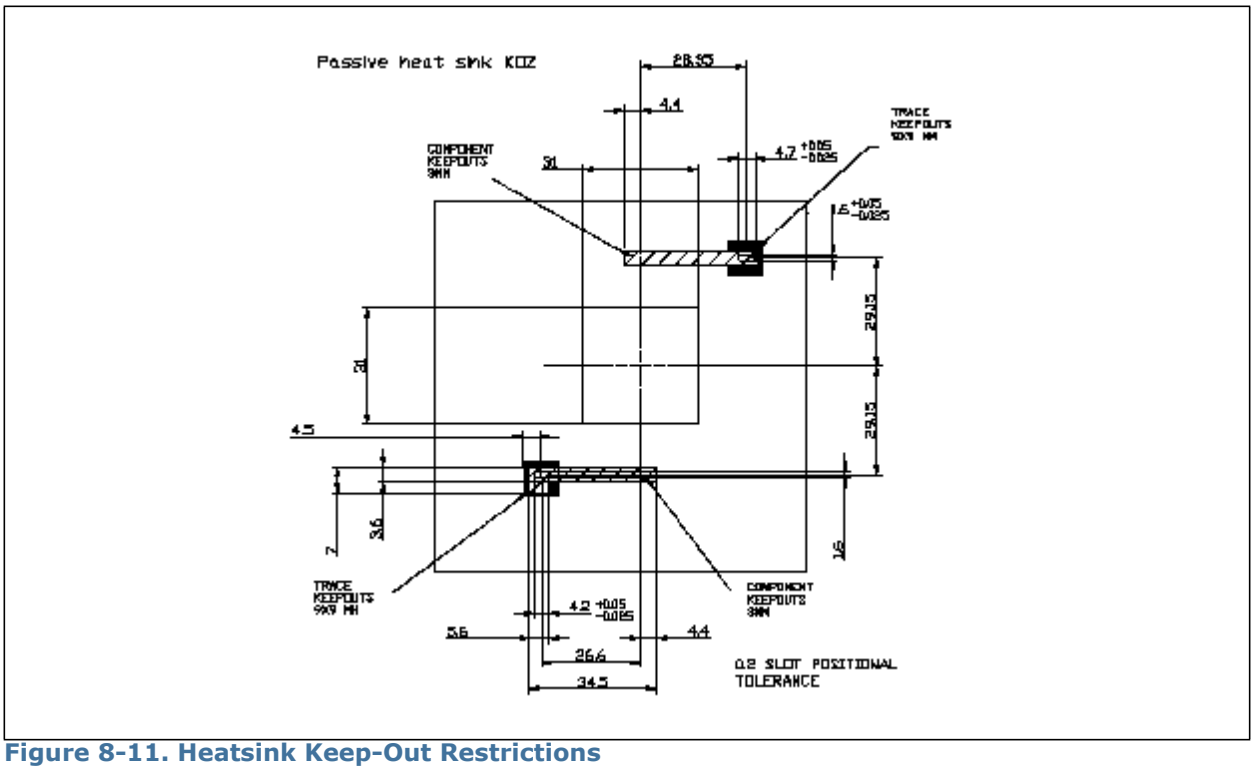


Figure 8-11. Heatsink Keep-Out Restrictions

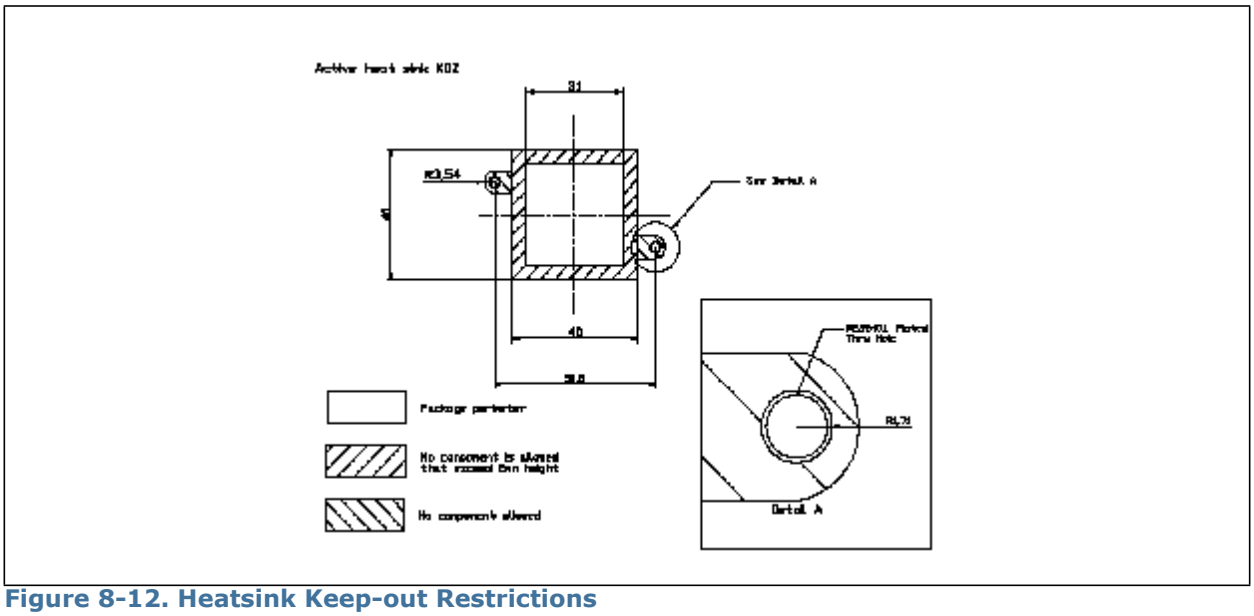


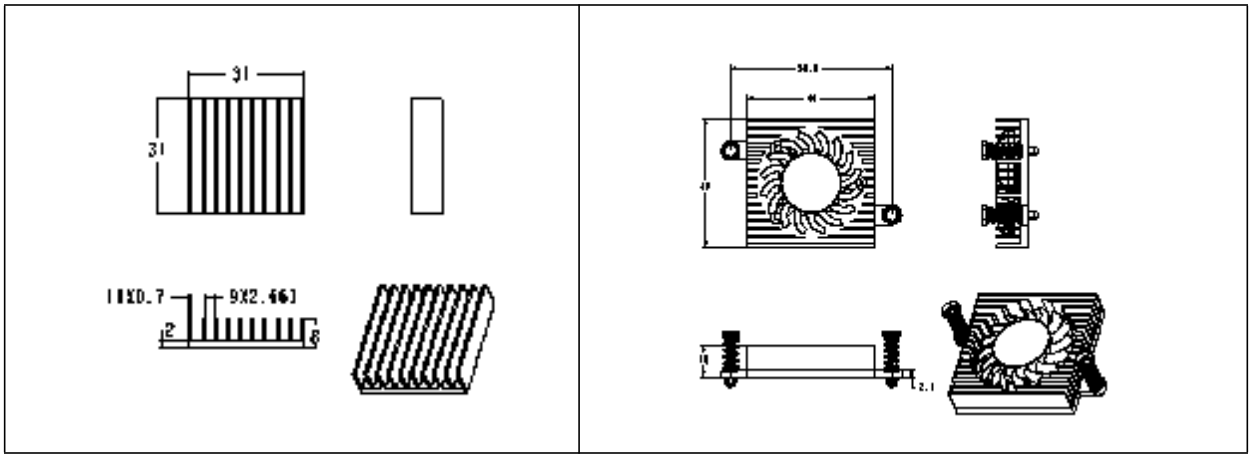
Figure 8-12. Heatsink Keep-out Restrictions

8.16.5.2 Default Enhanced Thermal Solution

If you have no control over the end-user's thermal environment, or if you wish to bypass the thermal modeling and evaluation process, use the Default Enhanced Thermal Solutions. This solution replicates the performance defined at the Thermal Design Power (TDP). If, after implementing the Recommended Enhanced Thermal Solution, the case temperature continues to exceed allowable values, then additional cooling is needed. This additional cooling may be achieved by improving airflow to the component and/or adding additional thermal enhancements. A 40x40x10mm active heat sink is an alternative to the passive heat sink.

8.16.5.3 Extruded Heatsinks

If required, the following extruded heatsinks are the suggested. Figure 8-13 shows a passive and an active heatsink drawing.



Extruded Aluminum Heat Sink ECB-00181-01-GP Fansink

Figure 8-13. Extruded Heatsinks

8.16.5.3.1 Attaching the Extruded Heatsink

The extruded heatsink may be attached using clips with a phase change thermal interface material.

8.16.5.3.1.1 Clips

A well-designed clip, in conjunction with a thermal interface material (tape, grease, etc.) often offers the best combination of mechanical stability. Use of a clip requires significant advance planning as mounting holes are required in the PCB. Use non-plated mounting with a grounded annular ring on the solder side of the board surrounding the hole. For a typical low-cost clip, set the annular ring inner diameter to 150 mils and an outer diameter to 300 mils. Define the ring to have at least eight ground connections. Set the solder mask opening for these holes with a radius of 300 mils.

8.16.5.3.1.2 Thermal Interface (PCM45 Series)

The PCM45 series from Honeywell* is recommended. These thermal interface pads are phase change materials formulated for use in high performance devices requiring minimum thermal resistance. They consist of an electrically non-conductive, dry film that softens at device operating temperatures resulting in greasy-like performance.

Each PCA, system and heatsink combination varies in attach strength. Carefully evaluate the reliability of tape attaches prior to high-volume use.

8.16.5.4 Thermal Considerations for Board Design

The following PCB design guidelines are recommended to maximize the thermal performance of FCBGA 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 82598 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 the air flow or supply excessively heated air.

IcePak* and FlowTherm* models are available; contact your Intel representative for information.

8.16.5.4.1 Reliability

Each PCA, system and heatsink combination varies in attach strength and long-term adhesive performance. Evaluate the reliability of the completed assembly prior to high-volume use. Reliability recommendations are shown in Table 8-11.

Table 8-11. Reliability Validation

Test1	Requirement	Pass/Fail Criteria2
Mechanical Shock	50G trapezoidal, board level 11 ms, 3 shocks/axis	Visual and Electrical Check
Random Vibration	7.3G, board level 45 minutes/axis, 50 to 2000 Hz	Visual and Electrical Check
High-TemperatureLife	85 °C 2000 hours total Checkpoints occur at 168, 500, 1000, and 2000 hours	Visual and Mechanical Check
Thermal Cycling	Per-Target Environment (for example: -40 °C to +85 °C) 500 Cycles	Visual and Mechanical Check
Humidity	85% relative humidity 85 °C, 1000 hours	Visual and Mechanical Check

1. Performed the above tests on a sample size of at least 12 assemblies from 3 lots of material (total = 36 assemblies).
2. Additional pass/fail criteria can be added at your discretion.

8.16.5.5 Thermal Interface Management for Heat-Sink Solutions

To optimize heatsink design, it is important to understand the interface between the heat spreader and the heatsink base. Thermal conductivity effectiveness depends on the following:

- Bond line thickness
- Interface material area
- Interface material thermal conductivity

8.16.5.5.1 Bond Line Management

The gap between the heat spreader and the heatsink base impacts heat-sink solution performance. The larger the gap between the two surfaces, the greater the thermal resistance. The thickness of the gap is determined by the flatness of both the heatsink base and the heat spreader, plus the thickness of the thermal interface material (for example, PSA, thermal grease, epoxy) used to join the two surfaces.

8.16.5.5.2 Interface Material Performance

The following factors impact the performance of the interface material between the heat spreader and the heatsink base:

- Thermal resistance of the material
- Wetting/filling characteristics of the material

8.16.5.5.3 Thermal Resistance of the Material

Thermal resistance describes the ability of the thermal interface material to transfer heat from one surface to another. The higher the thermal resistance, the less efficient the heat transfer. The thermal resistance of the interface material has a significant impact on the thermal performance of the overall thermal solution. The higher the thermal resistance, the larger the temperature drop required across the interface.

8.16.5.5.4 Wetting/Filling Characteristics of the Material

The wetting/filling characteristic of the thermal interface material is its ability to fill the gap between the heat spreader top surface and the heatsink. Since air is an extremely poor thermal conductor, the more completely the interface material fills the gaps, the lower the temperature-drop across the interface, increasing the efficiency of the thermal solution.

8.16.6 Measurements for Thermal Specifications

Determining the thermal properties of the system requires careful case temperature measurements. This chapter provides guidelines for doing accurate measurements.

8.16.6.1 Case Temperature Measurements

Special care is required when measuring the Tcase temperature to ensure an accurate temperature measurement. Use the following guidelines when making Tcase measurements:

- Measure the surface temperature of the case in the geometric center of the case top.
- Calibrate the thermocouples used to measure Tcase before making temperature measurements.
- Use 36-gauge (maximum) K-type thermocouples.

Care must be taken to avoid introducing errors into the measurements when measuring a surface temperature that is a different temperature from the surrounding local ambient air. Measurement errors may be due to a poor thermal contact between the thermocouple junction and the surface of the package, heat loss by radiation, convection, conduction through thermocouple leads, and/or contact between the thermocouple cement and the heat-sink base (if used).

8.16.6.2 Attaching the Thermocouple (No Heatsink)

The following approach is recommended to minimize measurement errors for attaching the thermocouple with no heatsink:

- Use 36-gauge or smaller-diameter K-type thermocouples.
- Ensure that the thermocouple has been properly calibrated.
- Attach the thermocouple bead or junction to the top surface of the package (case) in the center of the heat spreader using high thermal conductivity cements.

It is critical that the entire thermocouple lead be butted tightly to the heat spreader.

Attach the thermocouple at a 0° angle if there is no interference with the thermocouple attach location or leads (Figure 8-14). This method is recommended for use with packages not having a heat sink.

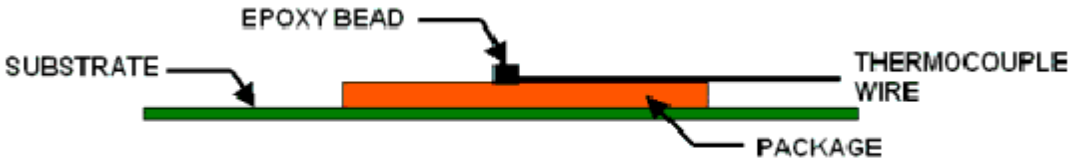


Figure 8-14. Technique for Measuring Tcase with 0° Angle Attachment, No Heatsink

8.16.6.3 Attaching the Thermocouple (Heatsink)

The following approach is recommended to minimize measurement errors for attaching the thermocouple with heatsink:

- Use 36-gauge or smaller diameter K-type thermocouples.
- Ensure that the thermocouple is properly calibrated.
- Attach the thermocouple bead or junction to the case's top surface in the geometric center using a high thermal conductivity cement. It is critical that the entire thermocouple lead be butted tightly against the case.
- Attach the thermocouple at a 90° angle if there is no interference with the thermocouple attach location or leads. This is the preferred method and is recommended for use with packages with heatsinks.
- For testing purposes, a hole (no larger than 0.150" in diameter) must be drilled vertically through the center of the heatsink to route the thermocouple wires out.
- Ensure there is no contact between the thermocouple cement and heatsink base. Any contact affects the thermocouple reading.

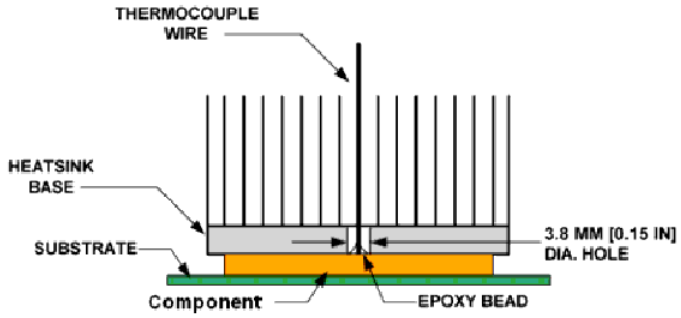


Figure 8-15. Technique for Measuring Tcase with 90° Angle Attachment

8.16.7 Heatsink and Attach Suppliers

Table 8-12. Heatsink and Attach Suppliers

Part	Part Number	Supplier	Contact
Extruded Al Heat sink + Clip + PCM45 (TIM) Assembly	Generated specific to customer numbering scheme	Cooler Master	Wendy Lin Cooler Master USA Inc., NJ office 603 First Ave., Unit 2C Raritan, NJ, 08869, USA Tel: 1-908-252-9400 Fax: 1-908-252-9299
Active fansink + pin + spring + PCM45 (TIM) Assembly	ECB-00181-01-GP	Cooler Master	Wendy Lin Cooler Master USA Inc., NJ office 603 First Ave., Unit 2C Raritan, NJ, 08869, USA Tel: 1-908-252-9400 Fax: 1-908-252-9299
PCM45 Series	PCM45F	Honeywell	North America Technical Contact: Paula Knoll 1349 Moffett Park Dr. Sunnyvale, CA 94089 Business: 858-279-2956

8.16.8 PHY Suppliers

Table 8-13. PHY Suppliers

Interface	Part Number	Supplier	Contact
XFP (LR/SR)	QT2022	Applied Micro Circuits Corporation (AMCC)	AMCC Sales Corporation - 800-935-AMCC (2622) 215 Moffett Park Drive Sunnyvale, CA 94089
SFP +	AEL2005	Netlogic	1NetLogic Microsystems - 650-961-6676 1875 Charleston Rd. Mountain View, CA 94043-1215
RJ45 (10BASE-T)	TN1010	Teranetics	Teranetics - 408-653-2200 3965 Freedom Circle 6th floor Santa Clara, CA 95054

§ §

9. Diagnostic Registers

9.1 Register Summary

Category	Offset	Abbreviation	Name	RW	Block (Internal Use Only)	Link
Diagnostic	0x110B0	GHECCR	GHOST ECC register	RW	GIO	

9.1.1 GHOST ECC Register - GHECCR (0x110B0, RW)

Undeclared bits are reserved.

Field	Bit(s)	Initial Value	Description
CDEM1EE	21	1	Completion Descriptor memory LAN 1 ECC enable.
CDAM1EE	18	1	Completion Data memory LAN 1 ECC enable.
CDEM0EE	9	1	Completion Descriptor memory LAN 0 ECC enable.
CDAM0EE	6	1	Completion Data memory LAN 0 ECC enable.

§ §



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

10. Models, Symbols, Testing Options, Schematics and Checklists

10.1 Models and Symbols

IBIS, BSDL, and HSPICE modeling data is available from Intel.

10.2 Physical Layer Conformance Testing

Physical layer conformance testing (also known as IEEE testing) is a fundamental capability for all companies with Ethernet LAN products. If your company does not have the resources and equipment to perform these tests, consider contracting the tests to an outside facility.

Once you integrate an external PHY with the 82598, the electrical performance of the solution should be characterized for conformance.

10.3 Schematics

Intel® 82598 10 GbE Controller Reference Schematics are available on Developer. See http://developer.intel.com/products/ethernet/index.htm?iid=nc+ethernet.

10.4 Checklists

Intel® 82598 10 GbE Controller Schematic and Layout and Placement Checklists are available on Developer. See http://developer.intel.com/products/ethernet/index.htm?iid=nc+ethernet.

§ §



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